



LabVIEW™

Manuel de référence des VIs et des fonctions

Filiales francophones

National Instruments Belgium nv Leuvensesteenweg 613 B-1930 Zaventem	National Instruments Canada–Montréal 1000 Boulevard St. Jean, Suite 316 Point-Claire, Québec H9R 5P1	National Instruments France Centre d’Affaires Paris-Nord Immeuble “Le Continental” BP 217 93153 Le Blanc-Mesnil Cedex	National Instruments Suisse Sonnenbergstr. 53 CH-5408 Ennetbaden
-------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------

Support internet

E-mail : support@natinst.com
Site FTP : <ftp.natinst.com>
Adresse web : <http://www.natinst.com>

Support Bulletin Board

BBS France : 01 48 65 15 59
BBS Etats-Unis : 512 794 5422

Support téléphonique en français

Belgique	Tél. : 02 757 00 20	Fax : 02 757 03 11	Tél. : 405 120 (Luxembourg)
Canada (Québec)	Tél. : 514 694 8521	Fax : 514 694 4399	
France	Tél. : 01 48 14 24 24	Fax : 01 48 14 24 14	
Suisse	Tél. : 056 200 51 51	Fax : 056 200 51 55	Tél. : 022 980 05 11 (Genève)

Les filiales

Allemagne 089 741 31 30, Australie 03 9879 5166, Autriche 0662 45 79 90 0, Brésil 011 288 3336,
Canada (Ontario) 905 785 0085, Corée 02 596 7456, Danemark 45 76 26 00, Espagne 91 640 0085,
Finlande 09 725 725 11, Hong Kong 2645 3186, Israël 03 6120092, Italie 02 413091, Japon 03 5472 2970,
Mexique 5 520 2635, Norvège 32 84 84 00, Pays-Bas 0348 433466, Royaume-Uni 01635 523545,
Singapour 2265886, Suède 08 730 49 70, Taiwan 02 377 1200

National Instruments Corporate Headquarters

11500 North Mopac Expressway Austin, Texas 78759 USA Tél: (+1) 512 794 0100

Informations importantes

Limitations de garantie

Les disquettes ou CD sur lesquels vous recevrez le logiciel de National Instruments sont garantis contre les pannes survenant lors de l'exécution de programmes, qui seraient dues à des défauts matériels ou de fabrication. La période de cette garantie est de 90 jours à partir de la date de livraison, attestée par les reçus ou autres documents. Le cas échéant, National Instruments corrigera ou remplacera la disquette ou le CD qui ne permettrait pas l'exécution normale des programmes, à condition qu'un tel défaut soit stipulé au cours de la période de garantie. National Instruments ne garantit pas que le fonctionnement de ses logiciels ne sera pas interrompu ni se déroulera sans erreur.

Un numéro RMA (Return Material Authorization) d'autorisation de retour de matériel doit être délivré par l'usine et clairement indiqué sur l'emballage du produit afin que celui-ci puisse être accepté pour la réparation sous garantie. National Instruments prendra à sa charge les frais de transport pour renvoyer à son propriétaire les éléments couverts par la garantie.

National Instruments estime avoir fait tout ce qu'il fallait pour que les informations contenues dans ce manuel soient exactes. Ce document a été soigneusement relu pour la précision des informations techniques qu'il contient. Au cas où il resterait malgré tout des erreurs techniques ou des fautes typographiques, National Instruments se réserve le droit d'apporter des modifications aux futures éditions de ce document sans préavis aux détenteurs de cette édition. Le lecteur est prié de prendre contact directement avec National Instruments s'il suspecte des erreurs. National Instruments ne pourra être tenu responsable des problèmes liés à l'utilisation de ce document ou aux informations qu'il contient.

À L'EXCEPTION DE CE QUI EST SPÉCIFIÉ ICI, NATIONAL INSTRUMENTS N'ACCORDE AUCUNE AUTRE GARANTIE, EXPLICITE OU IMPLICITE, ET REJETTE PARTICULIÈREMENT TOUTE GARANTIE LIÉE À L'ACTE DE VENTE ET À L'ADÉQUATION DE SES PRODUITS À UN BESOIN PARTICULIER. LES DROIT DES UTILISATEURS À RECOUVRIR LES DOMMAGES CAUSÉS PAR UNE FAUTE OU NÉGLIGENCE DE LA PART DE NATIONAL INSTRUMENTS SERONT LIMITÉS AUX SOMMES VERSÉES PAR L'UTILISATEUR. NATIONAL INSTRUMENTS NE SERA PAS PASSIBLE DE DOMMAGES ET INTÉRÊTS À LA SUITE DE PERTES DE DONNÉES OU DE PROFITS, OU DE DOMMAGES (ACCIDENTELS OU NON) LIÉS À L'UTILISATION DE SES PRODUITS, MÊME S'IL EN AVAIT ÉTÉ PRÉALABLEMENT AVERTI. Cette limitation de la responsabilité de National Instruments s'appliquera quelles que soient la nature et l'origine du préjudice, que ce soit à la suite d'un contrat ou la conséquence d'un acte délictueux, y compris par négligence. Toute action contre National Instruments doit être intentée dans l'année qui suit la cause de cette action. National Instruments ne pourra être tenu responsable de tout retard en performance dû à des causes qui iraient au-delà de ce qu'il lui est raisonnablement possible de faire. La garantie fournie ici ne couvre pas les dommages, défauts, dysfonctionnements ou défauts de service dus à des erreurs commises par l'utilisateur dans l'interprétation des instructions de National Instruments en ce qui concerne l'installation, le fonctionnement et la maintenance. Elle ne couvre pas non plus les négligences, les modifications ou mauvais usages du produit de la part de l'utilisateur, les chutes de tension ou les surtensions, le feu, les inondations, les accidents, les agissements de tierces personnes, et tout autre événement incontrôlable.

Copyright

Conformément à la loi sur les droits d'auteur, ce document ne peut être ni reproduit ni transmis, sous aucune forme que ce soit, informatique ou mécanique, notamment par photocopie, enregistrement, stockage dans un système d'archivage de documentation, ni traduit intégralement ou en partie, sans l'autorisation écrite de National Instruments Corporation.

Marques déposées

DAQ Card™, DAQ-STC™, DAQPad™, LabVIEW™, natinst.com™, National Instruments™, NI-DAQ™, PXI™, RTSI™ et SCXI™ sont des marques de National Instruments Corporation.

Les noms de produits et de sociétés cités sont des marques ou des marques déposées de leurs propriétaires respectifs.

MISE EN GARDE CONCERNANT L'UTILISATION DES PRODUITS DE National Instruments DANS LES APPLICATIONS MÉDICALES ET HOSPITALIÈRES

Les produits de National Instruments ne sont pas conçus avec des composants et suivant des méthodes de tests prévus pour assurer un niveau de fiabilité convenant à leur utilisation dans les applications de traitement et de diagnostic sur les personnes. Les applications des produits de National Instruments impliquant des traitements médicaux ou cliniques peuvent potentiellement occasionner des blessures accidentelles à cause d'une panne des produits, ou à cause d'une erreur de la part de l'utilisateur ou du concepteur de l'application. Toute utilisation ou application des produits National Instruments pour ou dans des traitements médicaux ou cliniques doit être effectuée par un personnel médical correctement formé et qualifié, et toutes les garanties médicales d'usage, tous les équipements et toutes les procédures qui sont appropriés à cette situation particulière pour éviter les blessures graves ou la mort, doivent toujours être mis en œuvre lorsque l'on utilise les produits de National Instruments. Les produits de National Instruments N'ONT PAS été conçus pour se substituer à toute forme de procédé, procédure ou équipement utilisés pour la surveillance médicale ou pour garantir la santé publique dans les traitements médicaux ou cliniques.

Table des matières

Avant-propos

Organisation de ce manuel.....	xxi
Conventions utilisées dans ce manuel	xxii
Références bibliographiques.....	xxiv
Communication avec l'utilisateur.....	xxiv

Chapitre 1

Introduction aux VIs et aux fonctions G

Repérer les fonctions G et les VIs	1-1
Vue d'ensemble des fonctions et des VIs	1-2
Structures	1-2
Fonctions numériques.....	1-3
Fonctions booléennes	1-3
Fonctions de chaîne de caractères	1-3
Fonctions de tableau	1-3
Fonctions de cluster.....	1-4
Fonctions de comparaison.....	1-4
Fonctions Temps et Dialogue.....	1-4
Fonctions d'E/S sur fichiers	1-4
Fonctions avancées.....	1-5
Acquisition de données (DAQ).....	1-5
E/S d'instruments	1-5
Communications.....	1-5
VIs d'analyse.....	1-6
Sélectionner un VI.....	1-6
Tutorial	1-6
Bibliothèque de drivers d'instruments.....	1-6
Bibliothèques utilisateur.....	1-7
Contrôle d'applications	1-7

PARTIE I

Fonctions G et VIs

Chapitre 2

Vue d'ensemble des références des VIs et des fonctions G

Vue d'ensemble des fonctions G	2-2
Introduction au polymorphisme	2-2
Polymorphisme	2-2
Polymorphisme d'unité	2-3
Conversion numérique	2-4
Débordement et sous-comptage	2-6
Styles de câblage	2-6

Chapitre 3

Structures

Vue d'ensemble des structures	3-2
-------------------------------------	-----

Chapitre 4

Fonctions numériques

Polymorphisme des fonctions numériques	4-2
Polymorphisme des fonctions transcendantes	4-4
Polymorphisme des fonctions de conversion	4-4
Polymorphisme des fonctions complexes	4-4
Descriptions des fonctions arithmétiques	4-4
Descriptions des fonctions de conversion	4-10
Descriptions des fonctions trigonométriques et hyperboliques	4-15
Descriptions des fonctions logarithmiques	4-18
Descriptions des fonctions complexes	4-21
Descriptions des constantes numériques supplémentaires	4-22

Chapitre 5

Fonctions booléennes

Polymorphisme des fonctions booléennes	5-2
Descriptions des fonctions booléennes	5-3

Chapitre 6

Fonctions de chaînes de caractères

Vue d'ensemble du polymorphisme des fonctions de chaînes de caractères.....	6-2
Polymorphisme des fonctions de chaînes de caractères	6-2
Polymorphisme des fonctions de conversion supplémentaire	6-2
Polymorphisme des fonctions de conversion de chaînes de caractères.....	6-2
Vue d'ensemble des chaînes de format.....	6-3
Descriptions des fonctions de chaînes de caractères.....	6-7
Descriptions des fonctions de conversion supplémentaires	6-18
Descriptions des fonctions de conversion de chaîne de caractères.....	6-22
Constantes de chaînes de caractères fixées	6-24

Chapitre 7

Fonctions de tableaux

Vue d'ensemble des fonctions de tableaux.....	7-2
Valeurs de l'indice de dépassement de gamme	7-3
Polymorphisme des fonctions de tableaux.....	7-3
Descriptions des fonctions de tableaux.....	7-3

Chapitre 8

Fonctions de clusters

Vue d'ensemble des fonctions de clusters	8-2
Polymorphisme des fonctions de clusters	8-3
Etablir l'ordre des éléments d'un cluster	8-3
Descriptions des fonctions de clusters	8-4

Chapitre 9

Fonctions de comparaison

Vue d'ensemble des fonctions de comparaison	9-2
Comparaison booléenne	9-2
Comparaison de chaînes de caractères	9-2
Comparaison numérique.....	9-2
Comparaison de clusters.....	9-2
Modes de comparaison	9-3
Comparaison de caractères	9-4
Polymorphisme des fonctions de comparaison.....	9-5
Descriptions des fonctions de comparaison.....	9-6

Chapitre 10

Fonctions de temps, de dialogue et d'erreurs

Vue d'ensemble des fonctions de temps, de dialogue et d'erreurs.....	10-2
Fonctions temporelles	10-2
Vue d'ensemble de la gestion d'erreurs	10-3
E/S d'erreur et cluster d'état d'erreur	10-4
Descriptions des fonctions de temps et de dialogue	10-7
Descriptions des VIs de gestion d'erreur.....	10-11

Chapitre 11

Fonctions des fichiers

Vue d'ensemble des fonctions et des VIs d'E/S sur fichiers.....	11-2
Les VIs de fichiers de haut niveau	11-2
VIs de bas niveau et fonctions de fichiers.....	11-2
Fichiers de structure d'octets et d'enregistrement de données	11-4
Paramètres transmissibles	11-5
E/S d'erreurs dans les fonctions d'E/S sur fichiers.....	11-5
Autorisations	11-6
Descriptions des VIs et des fonctions d'E/S sur fichiers.....	11-7
Descriptions des VIs de fichiers binaires	11-15
Descriptions des fonctions de fichiers avancées	11-16
VIs de fichiers de configuration	11-23
Descriptions des constantes de fichiers	11-29

Chapitre 12

Fonctions de contrôle d'applications

Fonctions de contrôle d'applications.....	12-2
Descriptions des fonctions d'aide.....	12-7
Fonctions de menus	12-8

Chapitre 13

Fonctions avancées

Descriptions des fonctions avancées	13-2
Descriptions des fonctions de manipulation de données.....	13-4
Descriptions des VIs de mémoire.....	13-8
VIs de synchronisation	13-9
VIs Notification	13-9
VIs File d'attente	13-12
VIs Rendez-vous	13-16

VIs Sémaphore.....	13-18
Descriptions des fonctions d'occurrences.....	13-21

PARTIE II

VIs d'acquisition de données

Chapitre 14

Introduction aux VIs d'acquisition de données LabVIEW

Aide en ligne pour les VIs DAQ.....	14-2
VIs d'entrées analogiques.....	14-4
VIs d'entrées analogiques simples.....	14-5
VIs d'entrées analogiques intermédiaires.....	14-5
Les VIs Utilitaires d'entrées analogiques.....	14-6
VIs d'entrées analogiques avancés.....	14-6
Exemples de VIs concernant les entrées analogiques.....	14-6
VIs de sorties analogiques.....	14-6
VIs de sorties analogiques simples.....	14-7
VIs de sorties analogiques intermédiaires.....	14-8
VIs Utilitaires de sorties analogiques.....	14-8
VIs de sorties analogiques avancés.....	14-9
Exemples de VIs concernant les sorties analogiques.....	14-9
VIs de fonctions numériques.....	14-9
VIs d'E/S numériques simples.....	14-10
VIs d'E/S numériques intermédiaires.....	14-10
VIs d'E/S numériques avancés.....	14-11
Exemples de VIs d'E/S numériques.....	14-11
VIs Compteurs.....	14-11
VIs Compteurs simples.....	14-12
VIs Compteurs intermédiaires.....	14-13
VIs Compteurs avancés.....	14-13
Exemples de VIs Compteurs.....	14-13
VIs d'étalonnage et de configuration.....	14-13
VIs de conditionnement de signaux.....	14-14

Chapitre 15

VIs d'entrées analogiques simples

Descriptions des VIs d'entrées analogiques simples.....	15-2
---------------------------------------------------------	------

Chapitre 16

VIs d'entrées analogiques intermédiaires

Gestion d'erreurs	16-1
Descriptions des VIs d'entrées analogiques intermédiaires	16-2

Chapitre 17

VIs Utilitaires d'entrées analogiques

Gestion d'erreurs	17-2
Descriptions des VIs Utilitaires d'entrées analogiques	17-2

Chapitre 18

VIs d'entrées analogiques avancés

Descriptions des VIs d'entrées analogiques avancés	18-2
----------------------------------------------------------	------

Chapitre 19

VIs de sorties analogiques simples

Descriptions des VIs de sorties analogiques simples	19-2
-----------------------------------------------------------	------

Chapitre 20

VIs de sorties analogiques intermédiaires

Gestion d'erreurs	20-1
Descriptions des VIs de sorties analogiques	20-2

Chapitre 21

VIs Utilitaires de sorties analogiques

Gestion d'erreurs	21-2
Descriptions des VIs Utilitaires de sorties analogiques	21-2

Chapitre 22

VIs de sorties analogiques avancés

Descriptions des VIs de sorties analogiques avancés	22-2
-----------------------------------------------------------	------

Chapitre 23

VIs d'E/S numériques simples

Description des VIs d'E/S numériques simples	23-2
----------------------------------------------------	------

Chapitre 24**VIs d'E/S numériques intermédiaires**

Gestion d'erreurs.....	24-2
Descriptions des VIs d'E/S numériques intermédiaires	24-3

Chapitre 25**VIs d'E/S numériques avancés**

Descriptions des VIs de port numérique.....	25-2
Descriptions des VIs de groupes numériques	25-3

Chapitre 26**VIs Compteurs simples**

Description des VIs Compteurs simples	26-2
---------------------------------------------	------

Chapitre 27**VIs Compteurs intermédiaires**

Gestion d'erreurs.....	27-2
Description des VIs Compteurs intermédiaires	27-2

Chapitre 28**VIs Compteurs avancés**

Description des VIs Compteurs avancés	28-2
---------------------------------------------	------

Chapitre 29**VIs d'étalonnage et de configuration**

Description des VIs d'étalonnage et de configuration.....	29-2
VIs de configuration de voies	29-20

Chapitre 30**VIs de conditionnement de signaux**

Description des VIs de conditionnement de signaux.....	30-2
--------------------------------------------------------	------

PARTIE III

Fonctions et VIs d'E/S d'instruments

Chapitre 31

Introduction aux VIs d'E/S d'instruments LabVIEW

Vue d'ensemble des drivers d'instruments.....	31-2
Distribution d'un driver d'instrument	31-3
Distribution d'un driver d'instrument sur CD-ROM.....	31-3
VIs de modèle de drivers d'instruments	31-4
Introduction à la bibliothèque VISA	31-4
Introduction au GPIB	31-5
Fonctions GPIB traditionnelles de LabVIEW	31-5
Fonctions GPIB 488.2.....	31-5
Fonctions avec périphérique unique	31-6
Fonctions avec plusieurs périphériques	31-6
Fonctions de gestion de bus.....	31-7
Fonctions de bas niveau.....	31-7
Fonctions générales	31-7
Vue d'ensemble des VIs de port série	31-7

Chapitre 32

Modèles de VIs de drivers d'instruments

Introduction aux modèles de VIs de drivers d'instruments.....	32-1
Descriptions des modèles de VIs de drivers d'instruments.....	32-2

Chapitre 33

Référence de la bibliothèque VISA

Opérations.....	33-2
Paramètres de référence de la bibliothèque VISA	33-2
Descriptions des opérations VISA.....	33-3
Fonctions de Gestion d'événements.....	33-10
Fonctions d'accès de haut niveau aux registres.....	33-12
Fonctions d'accès de bas niveau aux registres	33-16
Fonctions VISA série	33-18
Nœud de propriété VISA.....	33-19
Descriptions de Nœud de propriété VISA.....	33-20
Voie de données rapides	33-20
Paramètres généraux	33-20
Paramètres GPIB	33-21
Informations d'interface.....	33-21

Paramètres basés messages.....	33-21
Paramètres de la ligne modem.....	33-21
Ressources PXI.....	33-21
Paramètres PXI.....	33-21
Paramètres basés registres	33-22
Paramètres série.....	33-22
Informations sur la version.....	33-22
Paramètres VME/VXE	33-22

Chapitre 34

Fonctions GPIB traditionnelles

Paramètres des fonctions GPIB traditionnelles	34-2
Comportement des fonctions GPIB traditionnelles	34-3
Descriptions des fonctions GPIB traditionnelles	34-3
Fonctions d'appareil et de contrôleur GPIB	34-7
Fonctions d'appareil	34-8
Fonctions de contrôleur	34-9

Chapitre 35

Fonctions GPIB 488.2

Paramètres courants des fonctions GPIB 488.2.....	35-1
Descriptions des fonctions GPIB 488.2 (Fonctions d'appareil unique)	35-2
Descriptions des fonctions d'appareils multiples GPIB 488.2	35-4
Descriptions des fonctions de gestion du bus GPIB 488.2	35-7
Descriptions des fonctions d'E/S de bas niveau GPIB 488.2	35-10
Descriptions des fonctions générales GPIB 488.2.....	35-12

Chapitre 36

VIs de Port série

Descriptions des VIs Port série.....	36-1
--------------------------------------	------

PARTIE IV

VIs d'analyse

Chapitre 37

Introduction à l'analyse dans LabVIEW

Système de développement complet	37-2
Vue d'ensemble des VIs d'analyse.....	37-2
Organisation des VIs d'analyse	37-3
Nomenclature utilisée dans ce manuel	37-4

Chapitre 38

VIs de génération de signaux

Descriptions des VIs de génération de signaux	38-2
-----------------------------------------------------	------

Chapitre 39

VIs de traitement des signaux numériques

Description des VIs de traitement de signaux	39-2
----------------------------------------------------	------

Chapitre 40

VIs de mesure

Descriptions des VIs de mesure	40-2
--------------------------------------	------

Chapitre 41

VIs Filtres

Descriptions des VIs Filtres.....	41-2
-----------------------------------	------

Chapitre 42

VIs Fenêtres

Descriptions des VIs Fenêtres	42-1
-------------------------------------	------

Chapitre 43

VIs d'ajustement de courbe

Descriptions des VIs d'ajustement de courbe	43-2
---------------------------------------------------	------

Chapitre 44**VIs Probabilités et statistiques**

Descriptions des VIs Probabilités et statistiques44-2

Chapitre 45**VIs d'algèbre linéaire**

Descriptions des VIs d'algèbre linéaire45-2

Chapitre 46**VIs d'opérations sur tableaux**

Descriptions des VIs d'opérations sur tableaux.....46-2

Chapitre 47**VIs de méthodes numériques supplémentaires**

Descriptions des VIs de méthodes numériques supplémentaires47-2

PARTIE V**VIs et fonctions de communication****Chapitre 48****VIs TCP**

Description du VI TCP48-1

Fonctions TCP/IP.....48-2

Chapitre 49**VIs UDP**

Descriptions des VIs UDP49-1

Chapitre 50**VIs DDE**

Descriptions des VIs Client DDE50-2

Descriptions des VIs Serveur DDE50-4

Chapitre 51

Fonctions d'Automation ActiveX

Descriptions des fonctions d'Automation ActiveX.....	51-2
Fonctions de conversion de données	51-4

Chapitre 52

VIs AppleEvent

Comportement général des VIs AppleEvent	52-2
Boîte de dialogue d'identification de l'utilisateur.....	52-2
Identification de la destination	52-3
Options d'envoi.....	52-4
Descriptions des VIs de ciblage	52-4
Descriptions des VIs AppleEvent.....	52-6
VIs AppleEvent spécifiques à LabVIEW	52-8
Sujets avancés.....	52-9
Construire et envoyer d'autres AppleEvents	52-9
Créer des paramètres AppleEvent.....	52-10
VIs AppleEvent de niveau inférieur	52-14
Exemple de VI de support d'objets	52-17
Envoyer des AppleEvents à LabVIEW depuis d'autres applications	52-19
AppleEvents requis	52-19
AppleEvents spécifiques à LabVIEW.....	52-19
Réponses aux AppleEvents	52-20
Événement : Run VI	52-20
Description	52-20
Classe d'événement.....	52-20
ID de l'événement	52-20
Paramètres de l'événement.....	52-20
Paramètres de réponse.....	52-20
Erreurs possibles	52-21
Événement : Abort VI.....	52-21
Description	52-21
Classe d'événement.....	52-21
ID de l'événement	52-21
Paramètres de l'événement.....	52-21
Paramètres de réponse.....	52-22
Erreurs possibles	52-22

Événement : VI Active?	52-22
Description.....	52-22
Classe d'événement	52-22
ID de l'événement	52-22
Paramètres de l'événement	52-22
Paramètres de réponse	52-23
Erreurs possibles.....	52-23
Événement : Close VI	52-23
Description.....	52-23
Classe d'événement	52-23
ID de l'événement	52-23
Paramètres de l'événement.....	52-23
Paramètres de réponse	52-24
Erreurs possibles.....	52-24

Chapitre 53

VIs de communication entre programmes

Descriptions des VIs PPC	53-2
--------------------------------	------

Annexe A

Codes d'erreur

Codes d'erreur numériques	A-1
---------------------------------	-----

Annexe B

Capacités matérielles DAQ

Capacités matérielles des périphériques MIO et AI	B-2
Capacités matérielles des périphériques Série 1200/Lab et portables	B-12
Périphériques 54xx	B-16
Capacités logicielles des modules SCXI	B-19
Capacités matérielles des périphériques de sorties analogiques dédiés.....	B-24
Capacités matérielles des périphériques d'acquisition de signaux dynamiques.....	B-25
Capacités matérielles des périphériques numériques dédiés	B-27
Capacités matérielles des périphériques de séquençement dédiés.....	B-28
Capacités matérielles des périphériques 5102	B-29

Annexe C

Messages d'interface multiligne GPIB

Messages d'interface multiligne	C-1
Définitions des messages.....	C-6

Annexe D

Informations à l'attention du client

Index

Illustrations

Figure 27-1.	Mode de configuration dans “ICTR-Contrôler” (ICTR Control.vi).....	27-6
Figure 27-2.	Mode de configuration 1 dans “ICTR-Contrôler” (ICTR Control.vi).....	27-6
Figure 27-3.	Mode de configuration 2 dans “ICTR-Contrôler” (ICTR Control.vi).....	27-6
Figure 27-4.	Mode de configuration 3 dans “ICTR-Contrôler” (ICTR Control.vi).....	27-7
Figure 27-5.	Mode de configuration 4 dans “ICTR-Contrôler” (ICTR Control.vi).....	27-7
Figure 27-6.	Mode de configuration 5 dans “ICTR-Contrôler” (ICTR Control.vi).....	27-7
Figure 28-1.	Modes 2 et 3 de comptage non bufférisés.....	28-5
Figure 28-2.	Mode 3 de comptage bufférisé.....	28-5
Figure 28-3.	Mesure d'une largeur d'impulsion haute en mode 4 non bufférisé...	28-6
Figure 28-4.	Mesure d'une largeur d'impulsion de front montant en mode 4 bufférisé	28-6
Figure 28-5.	Mesure d'une période sur front montant en mode 4 non bufférisé ...	28-7
Figure 28-6.	Mesure d'une largeur d'impulsion sur front montant en mode 4 bufférisé	28-7
Figure 28-7.	Mesure d'une largeur d'impulsion haute en mode 6 non bufférisé...	28-8
Figure 28-8.	Mesure d'une largeur d'impulsion haute en mode 6 bufférisé (Décompte du front montant de la source).....	28-8
Figure 28-9.	Mesure d'une semi-période en mode 7 bufférisé	28-9
Figure 30-1.	Réseaux de commande par pont pour jauge de contrainte (Configuration quart de pont).....	30-4
Figure 30-2.	Réseaux de commande par pont pour jauge de contrainte (Configuration demi-pont)	30-5
Figure 30-3.	Réseaux de commande par pont pour jauge de contrainte (Configuration pont intégral)	30-6
Figure 30-4.	Diagramme du circuit d'une thermistance dans un diviseur de tension	30-7
Figure 30-5.	Diagramme du circuit d'une thermistance avec excitation de courant.....	30-8

Figure 41-1.	Filtre passe-bas	41-8
Figure 41-2.	Filtre passe-haut.....	41-8
Figure 41-3.	Filtre passe-bande	41-9
Figure 41-4.	Filtre coupe-bande	41-9

Tableaux

Tableau 6-1.	Codes spéciaux d'échappement.....	6-3
Tableau 6-2.	Syntaxe de chaînes de caractères.....	6-4
Tableau 6-3.	Erreurs possibles de la fonction "Formater en une chaîne de caractères"	6-8
Tableau 6-4.	Spécifications de format	6-9
Tableau 6-5.	Caractères spéciaux pour la recherche d'expressions.....	6-10
Tableau 6-6.	Exemples de chaînes de caractères pour la recherche d'expressions ..	6-12
Tableau 6-7.	Erreurs "Balayer une chaîne de caractères"	6-14
Tableau 6-8.	Exemples pour la fonction "Balayer une chaîne de caractères"	6-15
Tableau 9-1.	Descriptions des numéros de classe lexicale	9-9
Tableau 10-1.	Valeur valide des éléments du cluster Date/heure.....	10-2
Tableau 10-2.	Codes de format pour la chaîne de format de l'heure.....	10-8
Tableau 18-1.	Paramètres et gammes spécifiques au périphérique pour le VI "AI-Configurer un buffer" (AI Buffer Config.vi).....	18-3
Tableau 18-2.	Paramètres et gammes spécifiques aux périphériques pour le VI "AI-Configurer une horloge" (AI Clock Config.vi)	18-5
Tableau 18-3.	Paramètres et gammes spécifiques aux périphériques pour le VI "AI-Contrôler" (AI Control.vi)	18-7
Tableau 18-4.	Paramètres et gammes spécifiques aux périphériques pour le VI "AI-Configurer un groupe" (AI Group Config.vi).....	18-8
Tableau 18-5.	Configuration des voies pour "AI-Configurer un matériel" (AI Hardware Config.vi)	18-10
Tableau 18-6.	Paramètres et gammes spécifiques au périphérique pour le VI "AI-Configurer un matériel" (AI Hardware Config.vi).....	18-13
Tableau 18-7.	Paramètres et gammes spécifiques aux périphériques pour le VI "AI-BalayageUnique" (AI SingleScan.vi)	18-16
Tableau 18-8.	Restrictions pour déclenchement analogique sur les périphériques de Série E.....	18-19
Tableau 18-9.	Sources de déclenchement numérique pour des périphériques pourvus de sources de déclenchement numérique fixes	18-20
Tableau 18-10.	Paramètres et gammes spécifiques au périphérique pour le VI "AI-Configurer un déclenchement" (AI Trigger Config.vi)—Partie 1	18-21

Tableau 18-11.	Paramètres et gammes spécifiques au périphérique pour le VI “AI-Configurer un déclenchement” (AI Trigger Config.vi)—Partie 2	18-23
Tableau 18-12.	Paramètres et gammes spécifiques au périphérique pour le VI “AI-Configurer un déclenchement” (AI Trigger Config.vi)—Partie 3	18-24
Tableau 18-13.	Paramètres et gammes spécifiques au périphérique pour le VI “AI-Configurer un déclenchement” (AI Trigger Config.vi)—Partie 4	18-25
Tableau 25-1.	Paramètres et valeurs autorisées spécifiques aux périphériques	25-6
Tableau 28-1.	Circuits de compteurs et leurs périphériques DAQ disponibles	28-2
Tableau 28-2.	Numéros de compteurs pour les périphériques “CTR-Configurer un groupe” (CTR Group Config.vi).....	28-3
Tableau 28-3.	Compteurs adjacents	28-10
Tableau 29-1.	Exemples de paramètres pour les VIs “Voies en indice” (Channel to Index.vi)	29-8
Tableau 29-2.	Exemples de paramètres pour les VIs “Voies en indices” (Channel to Index.vi) sans Sun	29-9
Tableau 34-1.	Chaînes de commande des fonctions d’appareil	34-4
Tableau 34-2.	Chaînes de commande des fonctions du contrôleur	34-4
Tableau 51-1.	Anciennes et nouvelles fonctions d’Automation ActiveX.....	51-2
Tableau 52-1.	Formats des chaînes de caractères descriptives d’AppleEvent	52-12
Tableau A-1.	Gammes de codes d’erreur numériques	A-1
Tableau A-2.	Codes d’erreur VISA.....	A-2
Tableau A-3.	Codes d’erreur d’analyse.....	A-4
Tableau A-4.	Codes d’erreur des VIs d’acquisition de données	A-7
Tableau A-5.	Codes d’erreur AppleEvents	A-24
Tableau A-6.	Codes d’erreur des drivers d’instruments.....	A-25
Tableau A-7.	Codes d’erreur PPC.....	A-26
Tableau A-8.	Codes d’erreur GPIB.....	A-27
Tableau A-9.	Codes d’erreur des fonctions LabVIEW	A-28
Tableau A-10.	Codes d’erreur PPC spécifiques à LabVIEW	A-31
Tableau A-11.	Codes d’erreur TCP et UDP.....	A-31
Tableau A-12.	Codes d’erreur des ports série	A-32
Tableau A-13.	Codes d’erreur des messages AppleEvent spécifiques à LabVIEW ..	A-32
Tableau A-14.	Codes d’erreur DDE.....	A-32

Tableau B-1.	Programmabilité de la configuration d'entrées analogiques—Périphériques MIO et AI.....	B-2
Tableau B-2.	Caractéristiques des entrées analogiques—Périphériques MIO et AI (1 ^{ère} partie)	B-2
Tableau B-3.	Caractéristiques des entrées analogiques—Périphériques MIO et AI (2 ^{ème} partie)	B-4
Tableau B-4.	Support de voies internes—Périphériques MIO et AI.....	B-5
Tableau B-5.	Caractéristiques des sorties analogiques—Périphériques MIO et AI	B-5
Tableau B-6.	Caractéristiques des sorties analogiques—Périphériques Série E	B-9
Tableau B-7.	Capacités matérielles des E/S numériques—Périphériques MIO et AI	B-10
Tableau B-8.	Caractéristiques des compteurs—Périphériques MIO et AI.....	B-11
Tableau B-9.	Utilisation des compteurs pour entrées et sorties analogiques—Périphériques MIO et AI.....	B-12
Tableau B-10.	Programmabilité de la configuration des entrées analogiques—Périphériques Série 1200/Lab et portables	B-12
Tableau B-11.	Caractéristiques des entrées analogiques—Périphériques Série 1200/Lab et portables (1 ^{ère} partie)	B-13
Tableau B-12.	Caractéristiques des entrées analogiques—Périphériques Série 1200/Lab et portables (2 ^{ème} partie).....	B-14
Tableau B-13.	Caractères des sorties analogiques—Périphériques Série 1200/Lab et portables	B-14
Tableau B-14.	Usage des compteurs pour les entrées et les sorties analogiques—Périphériques Série 1200/Lab et portables	B-15
Tableau B-15.	Capacités matérielles des E/S numériques—Périphériques Série 1200/Lab et portables	B-15
Tableau B-16.	Caractéristiques des sorties analogiques et numériques—Périphériques Série 54XX	B-16
Tableau B-17.	Caractéristiques des compteurs/timers—Périphériques Série 1200/Lab et portables	B-18
Tableau B-18.	Caractéristiques des entrées analogiques—Modules SCXI (1 ^{ère} partie)	B-19
Tableau B-19.	Caractéristiques des sorties analogiques—Modules SCXI	B-20
Tableau B-20.	Caractéristiques des relais—Modules SCXI	B-20
Tableau B-21.	Caractéristiques des entrées et des sorties numériques—Modules SCXI.....	B-21
Tableau B-22.	Guide de sélection des blocs de terminaison—Modules SCXI	B-22
Tableau B-23.	Programmabilité de la configuration des entrées analogiques	B-23
Tableau B-24.	Programmabilité de la configuration des entrées analogiques	B-23
Tableau B-25.	Caractéristiques des sorties analogiques—Périphériques de sorties analogiques dédiés.....	B-24

Tableau B-26.	Programmabilité de la configuration des entrées analogiques— Périphériques d'acquisition de signaux dynamiques	B-25
Tableau B-27.	Caractéristiques des sorties analogiques— Périphériques d'acquisition de signaux.....	B-25
Tableau B-28.	Caractéristiques des entrées analogiques— Périphériques d'acquisition de signaux dynamiques	B-26
Tableau B-29.	Capacités matérielles numériques— Périphériques d'E/S numériques	B-27
Tableau B-30.	Capacités matérielles numériques— Périphériques de séquençement dédiés	B-28
Tableau B-31.	Caractéristiques des compteurs/timers— Périphériques de séquençement dédiés	B-28
Tableau B-32.	Programmabilité de la configuration des entrées analogiques	B-29
Tableau B-33.	Caractéristiques des entrées analogiques	B-29
Tableau B-34.	Caractéristiques des entrées analogiques, (2 ^{ème} partie).....	B-29

Avant-propos

Le *Manuel de référence des VIs et des fonctions de LabVIEW* décrit tous les instruments virtuels (VIs) et les fonctions, dont :

- Les VIs qui supportent des dispositifs d'acquisition de données
- Les VIs adaptés aux opérations GPIB, VXIbus et d'E/S sur le port série
- Les VIs numériques statistiques, de filtrage et de traitement des signaux numériques
- Les VIs de communication de réseau et entre applications

Ce manuel constitue un supplément du *Manuel de l'utilisateur LabVIEW* que vous devez avoir déjà lu.

Ce manuel fournit des informations générales sur chaque fonction et VI disponibles dans le système de développement LabVIEW. Néanmoins, vous pouvez obtenir des informations plus spécifiques sur les paramètres de chaque fonction et de chaque VI en consultant le système de référence en ligne (sélectionnez **Aide»Référence en ligne**) ou la fenêtre d'Aide (sélectionnez **Aide»Afficher Aide**).

Organisation de ce manuel

Ce manuel comprend cinq sujets : fonctions G et VIs, VIs d'acquisition de données, VIs d'E/S d'instrument, VIs d'analyse et VIs de communication. Le chapitre 1, *Introduction aux VIs et aux fonctions G*, présente les fonctions et les VIs disponibles dans le système de développement LabVIEW.

- La Partie I, *Fonctions G et VIs*, comprend les chapitres 2 à 13 et décrit les fonctions particulières au langage de programmation G.
- La Partie II, *VIs d'acquisition de données*, comprend les chapitres 14 à 30 et décrit les VIs d'acquisition de données (DAQ).
- La Partie III, *Fonctions et VIs d'E/S d'instruments*, comprend les chapitres 31 à 36 et décrit les VIs et les fonctions d'E/S d'instrument.
- La Partie IV, *VIs d'analyse*, comprend les chapitres 37 à 47 et décrit les VIs d'analyse.
- La Partie V, *VIs et fonctions de communication*, comprend les chapitres 48 à 53 et décrit les VIs de communication.

De plus, ce manuel comprend les annexes et l'index suivants :

- L'Annexe A, *Codes d'erreur*, comprend des tableaux qui résument les capacités d'E/S analogiques et numériques des dispositifs d'acquisition de données de National Instruments.
- L'Annexe B, *Capacités matérielles DAQ*, liste les commandes définies par IEEE 488.
- L'Annexe C, *Messages d'interface multiligne GPIB*, décrit les concepts de base permettant de comprendre le fonctionnement du GPIB.
- L'Annexe D, *Informations à l'attention du client*, comprend des formulaires pour vous aider à réunir les renseignements qui nous sont nécessaires pour résoudre vos problèmes techniques. Cette annexe comprend également un formulaire sur lequel vous pouvez écrire des remarques concernant la documentation du produit.
- L'*Index* présente une liste alphabétique des VIs décrits dans ce manuel et indique la page où vous pouvez les trouver.

Conventions utilisées dans ce manuel

Les conventions suivantes sont utilisées dans ce manuel :

< >

Les crochets à chevrons contiennent le nom d'une touche du clavier : par exemple, <Maj>. Les crochets à chevrons contenant des nombres séparés par des points de suspension représentent un ensemble de valeurs associé à un bit ou un nom de signal : par exemple, DBIO<3.0>.

[]

Les crochets droits représentent des éléments optionnels (par exemple, [réponse]).

-

Un trait d'union entre deux ou plusieurs noms de touches, dans des crochets à chevrons signifie que vous devez appuyer en même temps sur les touches désignées : par exemple, <Control-Alt-Supprimer>.

»

Le symbole » vous guide à travers des éléments du menu et des options de boîtes de dialogue imbriqués vers une action finale. La séquence **Fichier»Mise en page»Options»Fontes de substitution** vous invite à dérouler le menu **Fichier**, à sélectionner l'élément **Mise en page** puis **Options**, et enfin à sélectionner les options **Fontes de substitution** dans la dernière boîte de dialogue.

gras	Un texte en gras désigne des menus, des options de menu, des paramètres, des boîtes de dialogue, des boutons ou des options de boîte de dialogue, des icônes, des fenêtres, des onglets de Windows 95 ou des voyants lumineux.
<i>gras italique</i>	Cette icône associée à un texte en caractères gras et en italique signale une remarque, vous apportant des informations importantes.
monospace gras	Un texte en gras avec cette police de caractères désigne les messages et les réponses que l'ordinateur affiche automatiquement à l'écran. Cette fonte est également utilisée pour les lignes de code.
CTRL	Les noms de touche sont en majuscules.
<i>italique</i>	Un texte en italique désigne des variables, un élément important, une référence croisée ou la présentation d'une notion essentielle. Cette police désigne aussi un texte dont vous devez spécifier le mot ou la valeur correcte (par exemple, Windows 3.x).
<i>monospace italique</i>	Un texte en italique avec cette police de caractères indique que vous devez le remplacer par des mots ou des valeurs corrects.
monospace	Un texte avec cette police de caractères désigne du texte ou des caractères que vous devez littéralement taper, des sections de code, des exemples de programmation et des exemples de syntaxe. Cette police sert également aux noms propres des lecteurs, des chemins, des répertoires, des programmes, des sous-programmes, des sous-routines, des noms de dispositif, des fonctions, des opérations, des variables, des noms de fichier et des extensions, ainsi qu'aux messages et commentaires provenant des programmes.
chemins	Les chemins de ce manuel sont signalés par des barres obliques inverses (\) qui séparent des noms de lecteurs, de répertoires, de dossiers et de fichiers.

Références bibliographiques

Les documents suivants pourront vous être utiles lors de la lecture de ce manuel :

- *Manuel de l'utilisateur LabVIEW*
- *Manuel de référence de programmation en G*
- *Manuel de base d'acquisition de données de LabVIEW*
- *Guide de prise en main de LabVIEW*
- *Référence en ligne de LabVIEW, disponible en sélectionnant **Aide**»**Référence en ligne***
- *Tutoriel en ligne de LabVIEW (Windows uniquement), que vous lancez dans la boîte de dialogue LabVIEW*
- *Carte d'initiation LabVIEW*
- *Carte de référence rapide de programmation en G*
- *Notes d'information de LabVIEW*
- *Notes de mise à jour de LabVIEW*

Communication avec l'utilisateur

National Instruments aimerait recevoir vos commentaires sur ses produits et manuels. Les applications que vous développez avec nos produits nous intéressent et nous aimerions vous aider en cas de problèmes avec ces derniers. Pour que vous puissiez nous contacter facilement, ce manuel contient des formulaires de commentaires et de configuration que vous êtes invité à remplir. Vous trouverez ces formulaires à l'Annexe D, *Informations à l'attention du client*, à la fin de ce manuel.

Introduction aux VIs et aux fonctions G

Ce chapitre présente des informations de base sur les fonctions et les instruments virtuels (VIs) disponibles dans le système de développement LabVIEW.

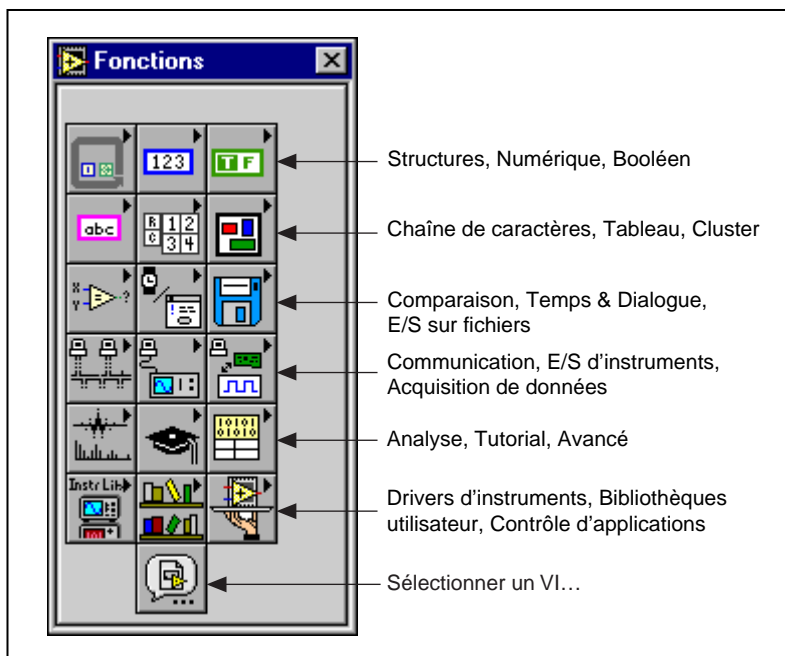
Le système de développement comprend des collections de VIs qui fonctionnent avec votre langage de programmation G, vos périphériques matériels d'acquisition de données (DAQ), vos périphériques d'instruments et autres interfaces de communication.

Repérer les fonctions G et les VIs

Les fonctions constituent des nœuds élémentaires dans le langage de programmation G. Elles sont comparables aux opérateurs ou aux fonctions des bibliothèques des langages conventionnels. Les fonctions ne sont pas des VIs et ne sont donc pas munies de faces-avant ni de diagrammes. Quand elles sont compilées, les fonctions génèrent un code machine en ligne.

Vous devez sélectionner les fonctions dans la palette **Fonctions** du diagramme. Quand la fenêtre du diagramme est activée, sélectionnez **Fenêtres»Palette de fonctions**. Vous pouvez également accéder à la palette **Fonctions** en ouvrant un menu local sur la partie de la fenêtre du diagramme où vous souhaitez poser la fonction.

L'illustration suivante présente les fonctions et les VIs disponibles dans la palette **Fonctions**.



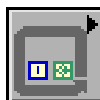
Plusieurs chapitres relatifs à la palette **Fonctions** comprennent des informations sur des exemples de fonctions. Les chemins de ces exemples LabVIEW commencent par la chaîne de caractères `exemples\`.

Vue d'ensemble des fonctions et des VIs

Les fonctions et VIs suivants sont disponibles dans la palette **Fonctions**.

Structures

Les structures en G comportent des structures Boucle While, Boucle For, Condition et Séquence. Cette palette comprend également les nœuds de variables locales et globales et la boîte de calcul.



Fonctions numériques

Les fonctions numériques sont utilisées pour réaliser des opérations mathématiques complexes, logarithmiques, trigonométriques, des conversions et des opérations arithmétiques. Cette palette comprend également des constantes numériques supplémentaires, comme π .



Fonctions booléennes

Les fonctions booléennes sont utilisées pour réaliser des opérations booléennes et logiques.



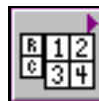
Fonctions de chaîne de caractères

Les fonctions de chaîne de caractères sont utilisées pour manipuler des chaînes de caractères et pour convertir des chaînes de caractères en nombre ou inversement. Cette palette comprend également des fonctions Chaîne vers Nombre supplémentaires et des fonctions de Conversion de chaîne.



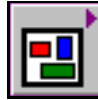
Fonctions de tableau

Les fonctions de tableau sont utilisées pour rassembler, séparer et traiter des tableaux.



Fonctions de cluster

Les fonctions de cluster sont utilisées pour rassembler, accéder ou modifier des éléments d'un cluster.



Fonctions de comparaison

Les fonctions de comparaison sont utilisées pour comparer des données (supérieures à, inférieures à, etc.) et réaliser des opérations basées sur une comparaison (par exemple, trouver les gammes minimales et maximales pour un groupe ou un tableau de valeurs).



Fonctions Temps et Dialogue

Les fonctions Temps et Dialogue sont utilisées pour manipuler les fonctions temporelles et l'affichage des boîtes de dialogue. Cette palette comprend également les VIs du traitement des erreurs.



Fonctions d'E/S sur fichiers

Les fonctions d'E/S sur fichiers sont utilisées pour manipuler les fichiers et répertoires. Cette palette comprend également les sous-palettes Fonctions de fichiers avancées, VIs de fichiers binaires et Constantes de fichiers.



Fonctions avancées

Les fonctions avancées sont des fonctions hautement spécialisées. Le Code Interface Node est un exemple de cette fonction avancée. La palette **Fonctions Avancées** comprend également les fonctions de contrôle de la fenêtre d'aide, les fonctions VI Serveur, les fonctions de manipulation de données et les fonctions d'occurrences.



Acquisition de données (DAQ)

Les VIs DAQ permettent de réaliser des opérations d'acquisition et de génération de données analogiques et numériques en temps réel, ainsi que des opérations de comptage. Reportez-vous au chapitre 14, [Introduction aux VIs d'acquisition de données LabVIEW](#), pour plus d'informations.



E/S d'instruments

Les VIs d'E/S d'instrument communiquent avec les instruments par le biais de communications GPIB, VISA ou série. Reportez-vous au chapitre 31, [Introduction aux VIs d'E/S d'instruments LabVIEW](#), pour plus d'informations.



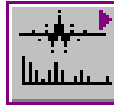
Communications

Les VIs de communication permettent la mise en réseau de plusieurs applications par le biais de TCP/IP, DDE, ActiveX, Apple Events, PPC ou UDP. Reportez-vous au chapitre 48, [Vis TCP](#), jusqu'au chapitre 53, [Vis de communication entre programmes](#), pour plus d'informations.



VIs d'analyse

Les VIs d'analyse permettent de mesurer et de générer des signaux, de traiter les signaux numériques, de les filtrer, de les fenêtrer, d'effectuer des opérations de probabilités et de statistiques, d'ajustement de courbe, d'algèbre linéaire et de tableau. Cette palette contient également des méthodes numériques supplémentaires. Reportez-vous au chapitre 37, [Introduction à l'analyse dans LabVIEW](#), pour plus d'informations.



Sélectionner un VI...

L'option **Sélectionner un VI...** permet de sélectionner et de poser sur le diagramme n'importe quel VI en utilisant une boîte de dialogue de fichier.



Tutorial

Les VIs tutoriaux fournissent des exemples que vous pouvez utiliser lors de la lecture du *Manuel de l'utilisateur LabVIEW*.



Bibliothèque de drivers d'instruments

Les drivers d'instruments constituent un ensemble de VIs utilisés pour les instruments GPIB, VISA, série et CAMAC. National Instruments, ainsi que d'autres revendeurs, distribuent ces drivers d'instruments. Tout driver que vous placez dans `instr.lib` apparaît dans la palette.



Bibliothèques utilisateur

La palette **Bibliothèques utilisateur** insère automatiquement tout VI contenu dans votre répertoire `user.lib`, facilitant l'accès aux sous-VIs les plus souvent utilisés que vous avez écrits.



Contrôle d'applications

La palette **Contrôle d'applications** comprend les fonctions d'aide, les fonctions de menu, les VIs d'impression et les VIs VI Serveur.



Fonctions G et VIs

La Partie I, *Fonctions G et VIs*, présente des descriptions sur les fonctions G et les VIs. Cette partie comprend les chapitres suivants :

- Le chapitre 2, *Vue d'ensemble des références des VIs et des fonctions G*, présente les fonctions G et les VIs. Ce chapitre décrit également les différences entre les fonctions et les VIs.
- Le chapitre 3, *Structures*, décrit les structures disponibles dans le G.
- Le chapitre 4, *Fonctions numériques*, décrit les fonctions à utiliser pour réaliser des opérations arithmétiques, complexes, logarithmiques et trigonométriques ainsi que des conversions. Il décrit également les constantes les plus communes comme la constante numérique, la constante énumérée et la constante menu déroulant, entre autres.
- Le chapitre 5, *Fonctions booléennes*, décrit les fonctions à utiliser pour réaliser des opérations logiques.
- Le chapitre 6, *Fonctions de chaînes de caractères*, décrit les fonctions de chaînes de caractères, notamment celles qui convertissent des chaînes de caractères en nombres et les nombres en chaînes de caractères.
- Le chapitre 7, *Fonctions de tableaux*, décrit les fonctions à utiliser pour les opérations sur les tableaux.
- Le chapitre 8, *Fonctions de clusters*, décrit les fonctions à utiliser pour les opérations sur les clusters.
- Le chapitre 9, *Fonctions de comparaison*, décrit les fonctions à utiliser pour réaliser des comparaisons ou des tests conditionnels.
- Le chapitre 10, *Fonctions de temps, de dialogue et d'erreurs*, décrit les fonctions temporelles, utilisées pour obtenir l'heure courante, mesurer le temps écoulé ou suspendre une opération pour une durée spécifique. La gestion d'erreurs est également discutée dans ce chapitre.

- Le chapitre 11, *Fonctions des fichiers*, décrit les VIs de bas niveau et les fonctions de manipulation des fichiers, des répertoires et des chemins. Ce chapitre décrit également les constantes de fichiers et les VIs de niveau supérieur.
- Le chapitre 12, *Fonctions de contrôle d'applications*, décrit les fonctions de contrôle d'application.
- Le chapitre 13, *Fonctions avancées*, décrit les fonctions utilisées pour exécuter des opérations avancées. Ce chapitre décrit également les fonctions d'aide, de manipulation de données et de synchronisation ainsi que les VIs de mémoire.

Vue d'ensemble des références des VIs et des fonctions G

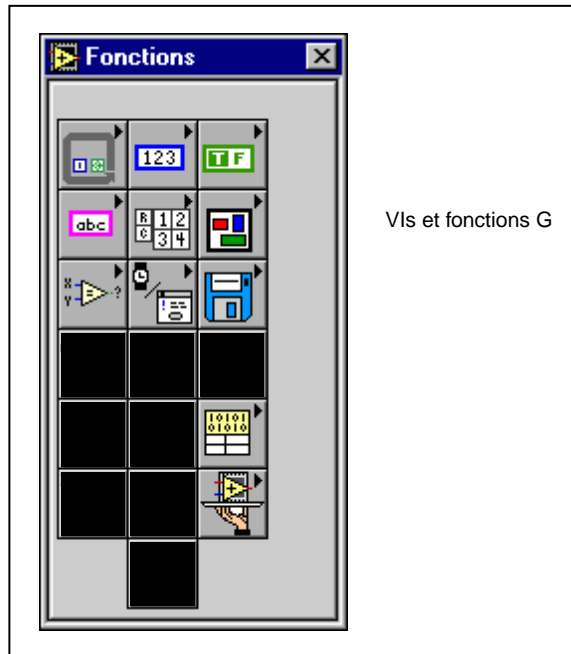
Ce chapitre présente les VIs et les fonctions G, dont les descriptions se trouvent dans les chapitres 3 à 13.

Les fonctions sont des nœuds élémentaires du langage de programmation G et sont comparables aux opérateurs ou aux bibliothèques de fonctions des langages conventionnels. Les fonctions ne sont pas des VIs et ne comportent donc pas de faces-avant ni de diagrammes. Une fois compilées, les fonctions génèrent un code machine.

Les VIs sont des “instruments virtuels” dont l'apparence est semblable à celle d'un instrument physique.

Vous devez sélectionner les fonctions G dans la palette **Fonctions**, à partir du diagramme. Quand la fenêtre du diagramme est activée, vous pouvez afficher la palette **Fonctions** en sélectionnant **Fenêtres»Palette de fonctions**. Vous pouvez également accéder à la palette **Fonctions** en ouvrant un menu local sur la partie de la fenêtre du diagramme où vous souhaitez poser la fonction.

L'illustration suivante présente les VIs et les fonctions G disponibles dans la palette **Fonctions**. Seuls les menus apparents sont traités dans cette partie. Pour avoir des informations sur les VIs et fonctions contenus dans les zones noircies, reportez-vous aux autres parties de ce manuel.



Plusieurs chapitres relatifs à la palette de fonctions présentent des informations sur les exemples de fonctions.

Vue d'ensemble des fonctions G

Reportez-vous au chapitre 1, *Introduction aux VIs et aux fonctions G* pour une description simple de chacune des onze palettes de VIs et de fonctions G disponibles.

Introduction au polymorphisme

Les sections suivantes présentent des informations générales sur le polymorphisme des fonctions G.

Polymorphisme

Le *polymorphisme* fait référence à l'aptitude d'une fonction à s'ajuster à des données d'entrée de représentations ou de types différents. La plupart des fonctions sont polymorphes. Les VIs, par contre, ne sont pas polymorphes. Toutes les fonctions comportant des entrées numériques acceptent n'importe quelle représentation numérique (sauf quelques fonctions qui n'acceptent pas de nombres complexes).

Les fonctions sont polymorphes à différents degrés ; aucune, quelques-unes ou toutes les entrées peuvent être polymorphes. Certaines entrées de fonction acceptent des nombres ou valeurs booléennes. Quelques-unes acceptent des nombres ou des chaînes de caractères tandis que d'autres acceptent non seulement des nombres scalaires mais aussi des tableaux de nombres, des clusters de nombres, des tableaux de clusters de nombres et ainsi de suite. D'autres encore n'acceptent que des tableaux à une dimension bien que les éléments du tableau puissent être de tout type. Certaines fonctions acceptent tous types de données, notamment les nombres complexes.

Polymorphisme d'unité

Si vous souhaitez créer un VI qui calcule une racine, ou la valeur quadratique moyenne d'une forme d'onde, vous devez définir l'unité associée à la forme d'onde. Un VI séparé est alors nécessaire pour les formes d'onde de tension, les formes d'onde de courant, les formes d'onde de température, etc. LabVIEW est pourvu d'une capacité d'unité polymorphe pour qu'un seul VI puisse effectuer le même calcul, sans tenir compte des unités reçues par les entrées.

Vous pouvez créer une unité polymorphe en entrant $\$x$, où x est un chiffre (par exemple, $\$1$). Considérez-la comme un point d'emplacement pour l'unité réelle. Quand LabVIEW invoque le VI, le programme remplace les unités que vous avez affectées à chaque occurrence de $\$x$ au sein de ce VI.

LabVIEW traite une unité polymorphe comme une unité unique. Vous ne pouvez pas convertir une unité polymorphe en une autre unité et les unités polymorphes se propagent à travers le diagramme comme toutes les autres unités. Quand l'unité est reliée à un indicateur qui contient également

L'abréviation §1, la correspondance est établie entre les unités et le VI peut être compilé.

Utilisez les combinaisons §1 comme des unités quelconques. Par exemple, si une entrée est multipliée par trois secondes, puis reliée à un indicateur, l'indicateur doit avoir des unités §1 s. Si des unités différentes sont affectées à un même indicateur, le diagramme affiche une mauvaise liaison. S'il vous faut plus d'une unité polymorphe, utilisez les abréviations §2, §3, etc.

Un appel à un sous-VI comprenant des unités polymorphes calcule les unités de sortie en fonction des unités reçues par ses entrées. Supposez, par exemple, que vous créez un VI pourvu de deux entrées avec les unités polymorphes §1 et §2 calculant une sortie sous la forme §1 §2 / s. Si un appel au VI reçoit des entrées avec l'unité m/s à l'entrée §1 et kg à l'entrée §2, LabVIEW calcule l'unité de sortie en $\text{kg m} / \text{s}^2$.

Supposez maintenant qu'un VI différent comprenne deux entrées en §1 et §1/s, et qu'il calcule une sortie en §1². Si un appel à ce VI reçoit des données en m/s à l'entrée §1 et en m/s^2 à l'entrée §1/s, LabVIEW calcule l'unité de sortie en m^2 / s^2 . Si ce VI reçoit des données en m à l'entrée §1 et en kg à l'entrée §1/s, LabVIEW déclare une des entrées en conflit d'unité et calcule (si possible) la sortie à partir de l'autre entrée.

Un VI polymorphe peut contenir un sous-VI polymorphe car LabVIEW fait la distinction entre les unités respectives.

Conversion numérique

Vous pouvez convertir n'importe quelle représentation numérique en toute autre représentation numérique des entrées numériques de représentations différentes à une fonction, celle-ci retournant en général une sortie dans le format le plus grand ou le plus large. Les fonctions convertissent les représentations plus petites en la représentation la plus large avant l'exécution.

Certaines fonctions, telles que Diviser, Sinus et Cosinus, produisent toujours une sortie à virgule flottante. Si vous reliez des entiers à leurs entrées, ces fonctions les convertissent en nombres double précision à virgule flottante avant d'exécuter le calcul.

Pour des quantités scalaires, à virgule flottante, il est conseillé d'utiliser des nombres double précision à virgule flottante. Les nombres simple précision à virgule flottante, économisent très peu de temps d'exécution, voire pas du tout, et provoquent des débordements bien plus facilement. Les

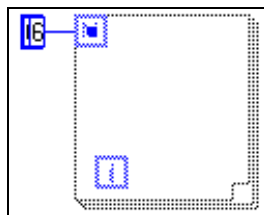
bibliothèques d'analyse, par exemple, utilisent des nombres double précision à virgule flottante. Vous ne devriez utiliser les nombres de précision étendue, à virgule flottante qu'en cas de nécessité. Les performances et la précision de l'arithmétique de précision étendue varient selon les plates-formes.

Pour les entiers, il est conseillé d'utiliser la représentation entier long.

Si vous reliez une sortie à une destination dont la représentation numérique diffère de la source, le G convertit les données conformément aux règles suivantes :

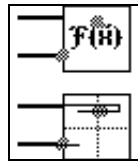
- Entier signé ou non signé en nombres à virgule flottante. La conversion est exacte, à l'exception des entiers longs en nombres simple précision à virgule flottante. Dans ce cas, le G réduit la précision de 32 bits à 24 bits.
- Nombre à virgule flottante en entier signé ou non signé. Le G ramène les valeurs hors gamme à la valeur minimale ou maximale de l'entier. La plupart des objets entiers, tels que le terminal d'itération d'une boucle For, arrondissent les nombres à virgule flottante. Le logiciel arrondit le nombre à une partie fractionnelle de 0,5 à l'entier pair le plus proche (par exemple, le G arrondit 6,5 à 6 au lieu de 7).
- Entier en entier. Le G ne ramène pas les valeurs hors gamme à la valeur minimale ou maximale de l'entier. Si la source est plus petite que la destination, le G étend le signe d'une source signée et met des zéros dans les bits supplémentaires d'une source non signée. Si la source est plus grande que la destination, le G ne copie que les bits d'ordre inférieur de la valeur.

Le diagramme met un *point de coercion* sur le bord d'un terminal pour indiquer qu'une conversion numérique automatique a eu lieu, comme le montre l'exemple suivant.



Vu que les VIs et les fonctions peuvent avoir plusieurs terminaux, un point de coercion peut apparaître à l'intérieur d'une icône si la liaison croise une

limite de terminal interne avant de quitter l'icône/connecteur, comme le montre la figure ci-dessous.



Déplacer une icône câblée étire un fil de liaison. Des points de coercion peuvent faire qu'un VI utilise plus de mémoire et mette plus de temps à s'exécuter. Essayez de maintenir des types de données compatibles avec vos VIs.

Débordement et sous-comptage

Le G ne vérifie pas les conditions de débordement et de sous-comptage des valeurs entières. Le débordement et le sous-comptage des nombres à virgule flottante sont conformes au standard IEEE 754 pour l'arithmétique binaire à virgule flottante.

Les opérations à virgule flottante propagent fidèlement NaN (Pas un Nombre) et \pm . Quand vous convertissez explicitement ou implicitement NaN ou \pm Inf en un entier ou en valeurs booléennes, vous obtenez cependant une valeur qui semble acceptable mais qui n'a aucune signification. Par exemple, diviser par zéro retourne \pm Inf, mais convertir cette valeur en un entier de type mots retourne une valeur de 32,768, soit la plus grande valeur pouvant être représentée dans le format de destination.

Styles de câblage

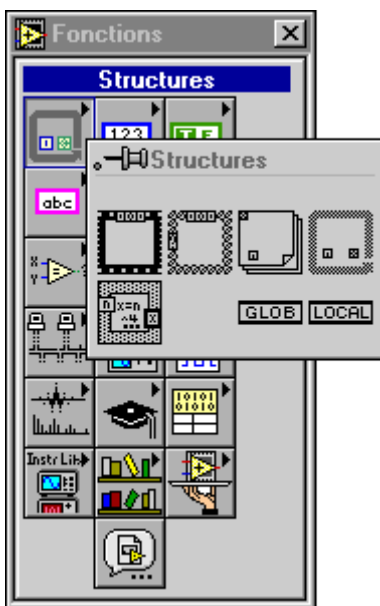
Le style de câblage représente les types de données de chaque terminal, comme l'indique le tableau ci-dessous. Les fonctions polymorphes affichent le style de câblage des types de données les plus communs.

	Scalaire	Tableau 1D	Tableau 2D	Tableau 3D	Tableau 4D
Nombre					
Booléen					
Chaîne de caractères					
Clusters généraux					
Clusters de nombres					

Structures

Ce chapitre décrit les structures disponibles en G.

Pour accéder à la palette **Structures**, sélectionnez **Fonctions»Structures**. L'illustration ci-dessous montre les options disponibles dans la palette **Structures**.



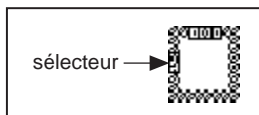
Consultez `examples\general\structs.llb` pour obtenir des exemples sur les différentes utilisations de ces structures dans LabVIEW.

Vue d'ensemble des structures

Les structures ci-dessous sont disponibles en G.

Structure Condition

Elle comprend un ou plusieurs sous-diagramme(s), ou *conditions*, dont une seule s'exécute. La condition qui s'exécute dépend du scalaire booléen, de la chaîne de caractères ou de la valeur numérique reliée au *sélecteur*.



Pour plus d'informations sur l'utilisation de la structure Condition dans LabVIEW, reportez-vous au chapitre 4, *Structure Condition, structure Séquence et boîte de calcul*, dans le *Manuel de l'utilisateur LabVIEW*.

Structure Séquence

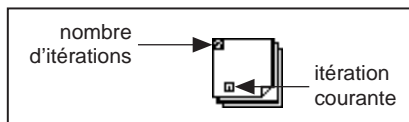
Elle comprend un ou plusieurs sous-diagramme(s), ou *étapes*, qui s'exécutent en séquences. Vous pouvez, en option, ajouter des variables locales de séquence pour transmettre des données d'une étape à d'autres en ouvrant un menu local sur le bord de la structure.



Pour plus d'informations sur l'utilisation de la structure Séquence dans LabVIEW, reportez-vous au chapitre 4, *Structure Condition, structure Séquence et boîte de calcul*, dans le *Manuel de l'utilisateur LabVIEW*.

Boucle For

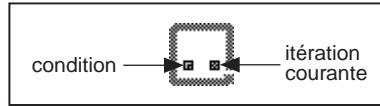
Elle exécute son sous-diagramme n fois, où n est égal à la valeur connectée au terminal de comptage. Vous pouvez, en option, ajouter des registres à décalage pour transmettre des données d'une itération à la suivante en ouvrant un menu local sur le bord de la structure.



Pour plus d'informations sur l'utilisation de la boucle For dans LabVIEW, reportez-vous au chapitre 3, *Boucles et graphes déroulants*, dans le *Manuel de l'utilisateur LabVIEW*.

Boucle While

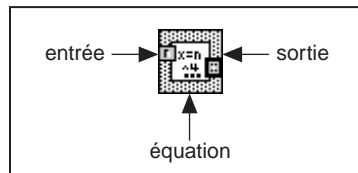
Elle exécute son sous-diagramme jusqu'à ce que la valeur booléenne reliée au *terminal conditionnel* soit fausse (FALSE). Vous pouvez, en option, ajouter des registres à décalage pour transmettre des données d'une itération à la suivante en ouvrant un menu local sur le bord de la structure.



Pour plus d'informations sur l'utilisation de la boucle While dans LabVIEW, reportez-vous au chapitre 3, *Boucles et graphes déroulants*, dans le *Manuel de l'utilisateur LabVIEW*.

Boîte de calcul

Elle exécute des formules mathématiques sur le diagramme.



Pour plus d'informations sur la boîte de calcul, reportez-vous au chapitre 4, *Structure Condition, structure Séquence et boîte de calcul*, dans le *Manuel de l'utilisateur LabVIEW*.

Variable globale

Il s'agit d'objet intégré de LabVIEW que vous devez définir en créant un type de VI particulier, pourvu de commandes de face-avant qui définissent le type de données de la variable globale. Vous pouvez écrire et lire des valeurs sur une variable globale.

GLOB

Pour plus d'informations sur les variables globales, reportez-vous au chapitre 23, *Variables globales et locales*, dans le *Manuel de référence de programmation en G*.

Variable locale

Elle vous permet de lire ou d'écrire sur l'un des indicateurs ou commandes de la face-avant de votre VI. Ecrire sur une variable locale produit le même résultat que transmettre des données à un terminal, sauf que vous pouvez y écrire même si c'est une commande, ou le lire même si c'est un indicateur.

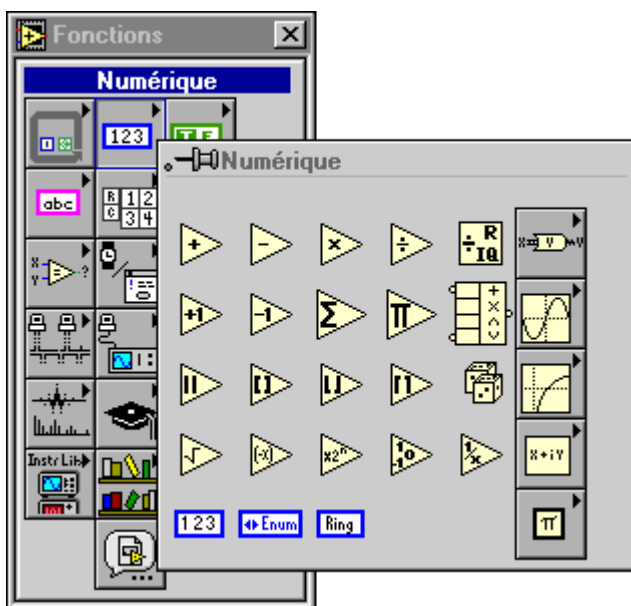
LOCAL

Pour plus d'informations sur les variables locales, reportez-vous au chapitre 23, *Variables globales et locales*, dans le *Manuel de référence de programmation en G*.

Fonctions numériques

Ce chapitre décrit les fonctions utilisées pour réaliser des opérations arithmétiques, complexes, logarithmiques et trigonométriques ainsi que de conversion. Il décrit également les constantes utilisées fréquemment telles que la constante numérique, la constante énumérée, la constante de type menu déroulant et les constantes numériques supplémentaires.

Pour accéder à la palette **Numérique**, sélectionnez **Fonctions»Numérique**. L'illustration ci-dessous montre les options disponibles sur la palette **Numérique**.



La palette **Numérique** comprend les sous-palettes suivantes :

- Constantes numériques supplémentaires
- Complexe
- Conversion
- Logarithmique
- Trigonométrique

Pour obtenir des exemples sur quelques-unes des fonctions arithmétiques, reportez-vous à `examples\general\structs.llb`.

Polymorphisme des fonctions numériques

Les fonctions arithmétiques acceptent des données d'entrée numériques. A quelques exceptions près indiquées dans les descriptions des fonctions, la sortie a la même représentation numérique que les entrées. Cependant si les entrées ont des représentations différentes, la représentation de la sortie correspond à l'entrée la plus large.

Les fonctions arithmétiques agissent sur des nombres, des tableaux de nombres, des clusters de nombres, des tableaux de clusters de nombres, des nombres complexes, et ainsi de suite. Une définition formelle et récursive du type d'entrée possible est la suivante :

type numérique = scalaire numérique || tableau numérique [*type numérique*] || cluster numérique [*types numériques*]

Les scalaires numériques peuvent être des nombres à virgule flottante, entiers ou complexes. Le G ne permet pas d'utiliser des tableaux de tableaux.

Les tableaux peuvent être de n'importe quelle dimension de n'importe quelle taille. Les clusters peuvent avoir n'importe quel nombre d'éléments. Les fonctions d'entrée unique agissent sur chaque élément de ces structures.

Pour les fonctions avec deux entrées, vous pouvez utiliser les combinaisons d'entrées suivantes :

- *Similaire* : les deux entrées partagent la même structure et la sortie a la même structure que les entrées.
- *Un scalaire* : une entrée est un scalaire numérique, l'autre est un tableau ou un cluster et la sortie est un tableau ou un cluster.

- *Tableau de* : une entrée est un tableau numérique, l'autre est le type numérique et la sortie est un tableau.

Lorsque les entrées sont similaires, le G exécute la fonction sur les éléments respectifs des structures. Le G peut par exemple ajouter deux tableaux élément-par-élément. Les deux tableaux doivent partager la même dimensionnalité. Vous pouvez ajouter des tableaux contenant un nombre différent d'éléments ; la sortie d'une telle addition comporte le même nombre d'éléments que la plus petite entrée. Les clusters doivent comporter le même nombre d'éléments et les éléments respectifs doivent avoir la même structure.



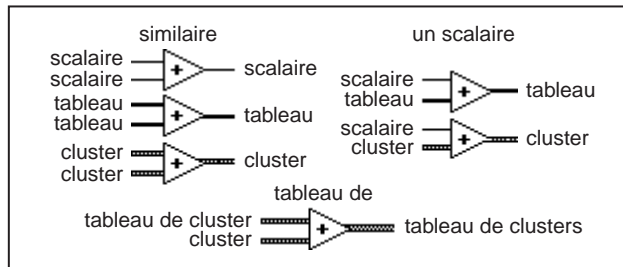
Remarque

Vous ne pouvez pas utiliser la fonction Multiplier pour des multiplications de matrices. Si vous utilisez la fonction Multiplier avec deux matrices, le G sélectionne le premier nombre dans la première rangée de la première matrice, le multiplie par le premier nombre de la première rangée de la seconde matrice, et ainsi de suite.

Pour des opérations impliquant un scalaire et un tableau ou un cluster, le G exécute la fonction sur le scalaire et les éléments respectifs de la structure. Par exemple, le G peut soustraire un chiffre de tous les éléments d'un tableau, sans tenir compte de la dimensionnalité du tableau.

Pour des opérations impliquant un type numérique et un tableau du même type, le G exécute la fonction sur chaque élément du tableau. Par exemple, un graphe étant un tableau de points et un point étant un cluster de deux types numériques, x et y , pour effectuer l'offset d'un graphe de 5 unités sur l'axe x et 8 unités sur l'axe y , vous pouvez ajouter un point, (5,8), à tous les points du graphe.

L'exemple ci-dessous montre quelques combinaisons polymorphes possibles pour la fonction Ajouter.



Polymorphisme des fonctions transcendantales

Les fonctions transcendantales acceptent une valeur d'entrée numérique. Si l'entrée est un entier, la sortie est un nombre double précision à virgule flottante. Dans le cas contraire, la sortie a la même représentation numérique que l'entrée.

Ces fonctions agissent sur des nombres, des tableaux de nombres, des clusters de nombres, des tableaux de clusters de nombres, des nombres complexes, et ainsi de suite.

Polymorphisme des fonctions de conversion

Toutes les fonctions de conversion sont polymorphes, à l'exception des fonctions "Tableau d'octets en chaîne de caractères", "Chaîne de caractères en tableau d'octets", "Convertir une unité" et "Changer les bases d'unité". Ainsi, les fonctions polymorphes agissent sur les valeurs scalaires, les tableaux de scalaires, les clusters de scalaires, les tableaux de clusters de scalaires, et ainsi de suite. La sortie a la même représentation numérique que l'entrée mais le type de la sortie est différent du type de l'entrée.

Lorsque vous comparez des entiers signés et non signés et que l'entier signé est négatif, l'entier négatif est rendu positif avant la comparaison, vous n'obtenez donc pas le résultat prévu. Par exemple, si vous entrez -1 avec la représentation I32 sur une entrée et 5 avec la représentation U32 sur l'autre entrée, le résultat retourné indique que la valeur minimale est 5, car 5 est inférieur à 4294967295.

Polymorphisme des fonctions complexes

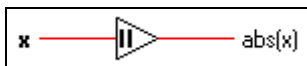
Les fonctions complexes agissent sur des valeurs scalaires, des tableaux de scalaires, des clusters de scalaires, des tableaux de clusters de scalaires, et ainsi de suite. La sortie a la même composition que l'entrée mais son type est différent.

Descriptions des fonctions arithmétiques

Les fonctions suivantes sont disponibles.

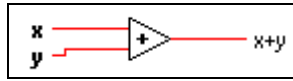
Valeur absolue

Retourne la valeur absolue de l'entrée.



Ajouter

Calcule la somme des entrées.



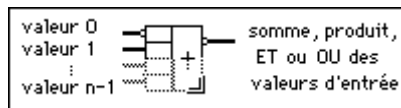
Sommer les éléments du tableau

Retourne la somme de tous les éléments contenus dans un **tableau numérique**.



Opérateur arithmétique

Exécute des opérations arithmétiques sur plusieurs entrées de type numérique, cluster ou booléen.



Vous choisissez l'opération (multiplier, ET ou OU) en ouvrant le menu local de la fonction et en sélectionnant **Changer le mode**.

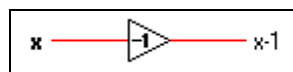
Vous pouvez inverser les entrées ou la sortie de cette fonction en ouvrant le menu local des terminaux séparément et en sélectionnant **Inverser**. Pour l'opération Ajouter, sélectionnez **Inverser** pour changer le signe d'une entrée ou de la sortie. Pour l'opération Multiplier, sélectionnez **Inverser** pour inverser une entrée ou la sortie. Pour l'opération ET ou OU, sélectionnez **Inverser** pour inverser logiquement une entrée ou la sortie.



Remarque *Vous pouvez ajouter des entrées à ce nœud en ouvrant le menu local sur une entrée et en sélectionnant "Ajouter une entrée" ou en positionnant l'outil Flèche sur le coin inférieur gauche ou droit et en le faisant glisser.*

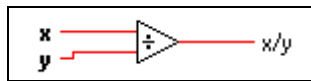
Décrément

Soustrait 1 à la valeur d'entrée.



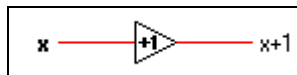
Diviser

Calcule le quotient des entrées.



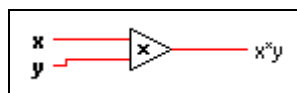
Incrément

Ajoute 1 à la valeur d'entrée.



Multiplier

Retourne le produit des entrées.



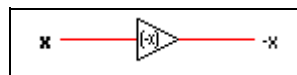
Multiplier les éléments du tableau

Retourne le produit de tous les éléments du **tableau numérique**.



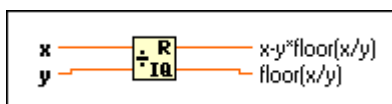
Opposer

Calcule l'opposé de la valeur d'entrée.



Quotient & Reste

Calcule le quotient entier et le reste des entrées.



Si y est une valeur entière égale à zéro, le quotient est égal à zéro et le reste est égal à x . Si y est un nombre flottant égal à zéro, le quotient est infini et le reste devient NaN.

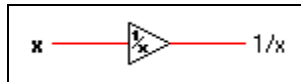
Nombre aléatoire (0–1)

Produit un nombre double précision à virgule flottante compris entre 0 et 1 exclusif (qui ne comprend pas 0 et 1). La distribution est uniforme.



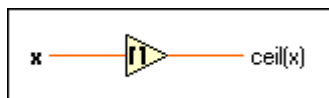
Inverser

Divise 1 par la valeur d'entrée.



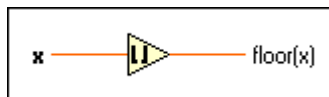
Arrondir à + l'infini

Arrondit l'entrée à l'entier supérieur le plus proche. Par exemple, si l'entrée est 3,1, le résultat est 4. Si l'entrée est -3,1, le résultat est -3.



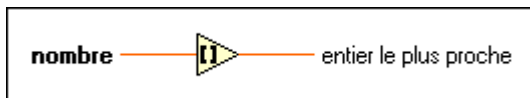
Arrondir à - l'infini

Arrondit l'entrée à l'entier inférieur le plus proche. Par exemple, si l'entrée est 3,8, le résultat est 3. Si l'entrée est -3,8, le résultat est -4.



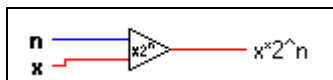
Arrondir à l'entier le plus proche

Arrondit l'entrée à l'entier le plus proche. Si la valeur de l'entrée est juste entre deux entiers (par exemple, 1,5 ou 2,5), la fonction retourne l'entier pair le plus proche (2).



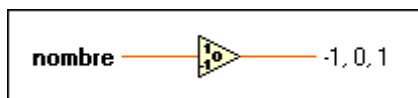
Multiplier par une puissance de 2

Multiplie une entrée (**x**) par 2 à la puissance de l'autre entrée (**n**). Si **n** est un nombre à virgule flottante, cette fonction arrondit **n** avant d'effectuer le calcul (0,5 est arrondi à 0 ; 0,51 est arrondi à 1). Si **x** est un entier, cette fonction est l'équivalent d'un décalage arithmétique.



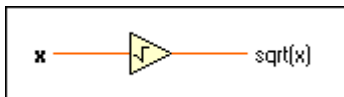
Signe

Retourne 1 si la valeur d'entrée est supérieure à 0, 0 si la valeur d'entrée est égale à 0 et -1 si la valeur d'entrée est inférieure à 0. Les autres langages de programmation nomment en général cette fonction `signum` ou `sgn`.



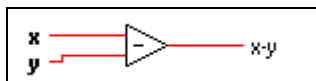
Racine carrée

Calcule la racine carrée de la valeur d'entrée. Si **x** est négatif et non complexe, la racine carrée est NaN.



Soustraire


Calcule la différence des entrées.



Constantes arithmétiques définies par l'utilisateur


Vous pouvez définir les constantes suivantes.

Constante numérique

-  Utilisez cette constante pour fournir une valeur numérique constante au diagramme. Définissez cette valeur en cliquant dans la constante avec l'outil Doigt et en tapant une valeur. Vous pouvez changer le format des données et leur représentation.


La valeur de la constante numérique ne peut pas être modifiée pendant l'exécution du VI. Vous pouvez affecter une étiquette à cette constante.

Constante énumérée

-  Les valeurs énumérées associent des entiers non signés à des chaînes de caractères. Si vous affichez une valeur au moyen d'une constante énumérée, la chaîne de caractères est affichée au lieu du nombre qui lui est associé. Si vous devez manipuler un ensemble de chaînes de caractères qui ne change pas, utilisez ce type de constante. Sélectionnez les valeurs en cliquant dans la constante avec l'outil Doigt. Définissez les chaînes de caractères avec l'outil Texte. Pour ajouter un autre élément, ouvrez le menu local de la constante et choisissez **Ajouter un élément avant** ou **Ajouter un élément après**.

La valeur de la constante énumérée ne peut pas être modifiée pendant l'exécution du VI. Vous pouvez affecter une étiquette à cette constante.

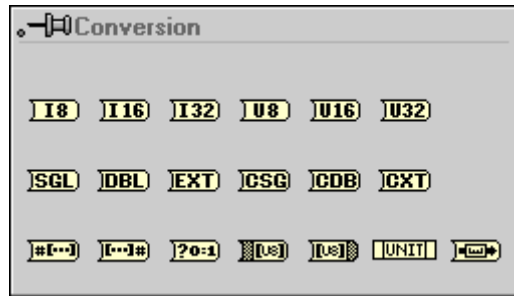
Constante de menu déroulant

-  Les menus déroulants associent des entiers non signés à des chaînes de caractères. Si vous affichez une valeur au moyen d'une constante de menu déroulant, le nombre est affiché au lieu de la chaîne de caractères qui lui est associée. Si vous devez manipuler un ensemble de chaînes de caractères qui ne change pas, utilisez cette constante. Sélectionnez les valeurs en cliquant dans la constante avec l'outil Doigt. Définissez les chaînes de caractères avec l'outil Texte. Pour ajouter un autre élément, ouvrez le menu local de la constante et choisissez **Ajouter un élément avant** ou **Ajouter un élément après**.

La valeur de la constante menu déroulant ne peut pas être modifiée pendant l'exécution du VI. Vous pouvez affecter une étiquette à cette constante.

Descriptions des fonctions de conversion

L'illustration suivante montre les options disponibles dans la sous-palette **Conversion**.



Les fonctions suivantes convertissent une entrée numérique en une représentation spécifique :

- En entier d'octets
- En complexe double précision
- En flottant double précision
- En complexe étendu
- En flottant précision étendue
- En entier long
- En complexe simple précision
- En flottant simple précision
- En entier d'octets non signé
- En entier mot non signé
- En entier long non signé
- En entier mot

Lorsque ces fonctions convertissent un nombre à virgule flottante en entier, elles l'arrondissent à l'entier le plus proche, ou à l'entier pair le plus proche si la partie fractionnelle est égale à 0,5. Si le résultat est hors gamme pour l'entier, ces fonctions retournent la valeur minimale ou maximale pour le type d'entier. Lorsque ces fonctions convertissent un entier en entier plus petit, elles copient les bits de moindre poids sans contrôler le débordement. Lorsque elles convertissent un entier en entier plus grand, elles étendent le signe d'un entier signé et comblent un entier non signé avec des zéros.

Faites attention pendant la conversion de nombres en représentations plus petites, en particulier lorsqu'il s'agit d'entiers, car les routines de conversion en G ne contrôlent pas le débordement.

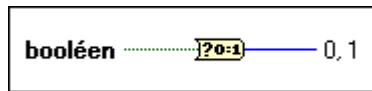
Tableau de booléens en nombre

Convertit un **tableau de booléens** en entier long non signé en l'interprétant comme une représentation en complément à deux de l'entier où l'élément 0 du tableau est le bit de moindre poids.



Booléen en (0,1)

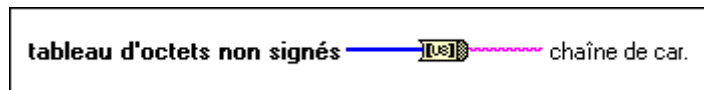
Convertit une valeur booléenne en entier de type mot : 0 et 1 pour les valeurs d'entrée fausse (FALSE) et vraie (TRUE), respectivement.



Booléen peut être un scalaire, un tableau ou un cluster de valeurs booléennes, un tableau de clusters de valeurs booléennes, et ainsi de suite. Reportez-vous à la section *Polymorphisme des fonctions booléennes* dans le chapitre 5, *Fonctions booléennes*.

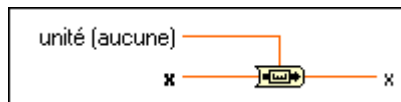
Tableau d'octets en chaîne de caractères

Convertit un tableau d'octets non signés en chaîne de caractères.



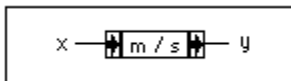
Changer les bases d'unité

Change les unités associées avec l'entrée en **unité** et retourne les résultats sur le terminal de sortie. Soyez extrêmement prudent quand vous employez cette fonction. Etant donné que la fonction **Changer les bases d'unité** fonctionne avec des bases, vous devez comprendre le principe de la conversion d'une unité arbitraire en ses bases avant de pouvoir bien utiliser cette fonction. Cette fonction peut changer des unités de base, telles que des mètres en grammes.



Convertir une unité

Convertit un nombre physique (un nombre muni d'une unité) en un nombre pur (un nombre sans unités) ou un nombre pur en un nombre physique.



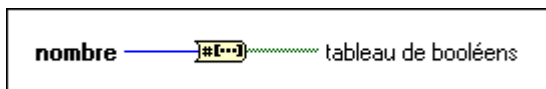
Vous pouvez modifier la chaîne de caractères de l'unité en la mettant en surbrillance avec l'outil Doigt et en l'éditant.

Si l'entrée est un nombre pur, la sortie reçoit les unités spécifiées. Si l'entrée est par exemple égale à 13 et la spécification d'unité est en secondes (s), la valeur résultante est 13 secondes.

Si l'entrée est un nombre physique, et **unité** est une unité compatible, la sortie est l'entrée exprimée en unités spécifiées. Par exemple, si vous spécifiez 37 mètres (m) et si **unité** est en mètres, le résultat est 37 sans unités associées. Si **unité** est en pieds (ft), le résultat est 121,36 sans unités associées.

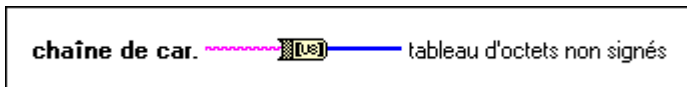
Nombre en tableau de booléens

Convertit un **nombre** entier en tableau de booléens de 8, 16 ou 32 éléments, où l'élément 0 correspond au bit de moindre poids de la représentation en complément à deux de l'entier.



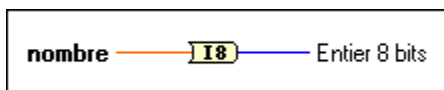
Chaîne de caractères en tableau d'octets

Convertit une **chaîne de caractères** en un tableau d'octets non signés.



En entier de type octet

Convertit un **nombre** en un entier 8 bits dans la gamme -128 à 127.



En complexe double précision

Convertit un **nombre** en nombre complexe double précision.



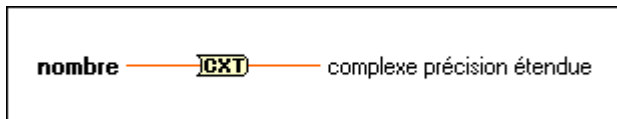
En flottant double précision

Convertit un **nombre** en nombre double précision à virgule flottante.



En complexe précision étendue

Convertit un **nombre** en nombre complexe à précision étendue.



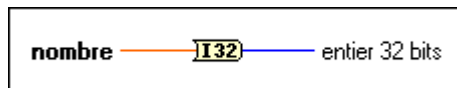
En flottant précision étendue

Convertit un **nombre** en nombre précision étendue à virgule flottante.



En entier long

Convertit un **nombre** en entier 32 bits dans la gamme -2^{31} à $2^{31}-1$.



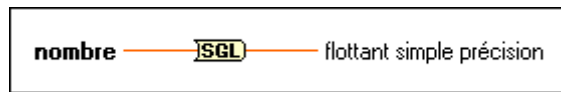
En complexe simple précision

Convertit un **nombre** en nombre complexe simple précision.



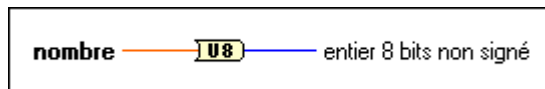
En flottant simple précision

Convertit un **nombre** en nombre simple précision à virgule flottante.



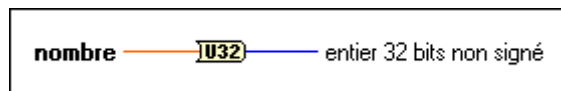
En entier octets non signé

Convertit un **nombre** en entier non signé 8 bits dans la gamme 0 à 255.



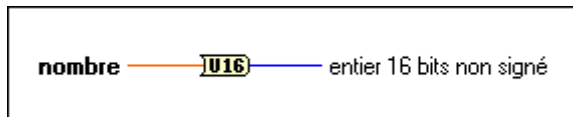
En entier long non signé

Convertit un **nombre** en entier non signé 32 bits dans la gamme 0 à $2^{32} - 1$.



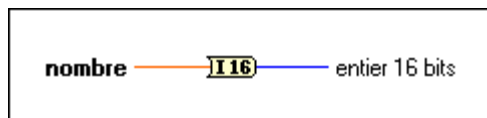
En entier de type mot non signé

Convertit un **nombre** en entier non signé 16 bits dans la gamme 0 à 65,535.



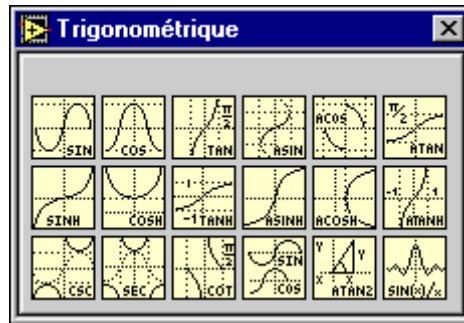
En entier de type mot

Convertit un **nombre** en entier 16 bits dans la gamme $-32,768$ à $32,767$.



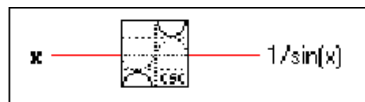
Descriptions des fonctions trigonométriques et hyperboliques

L'illustration ci-dessous montre les options de la sous-palette **Trigonométrique**.



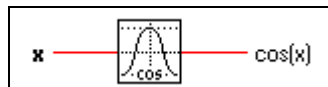
Cosécante

Calcule la cosécante de x , où x est en radians. La cosécante est l'inverse du sinus.



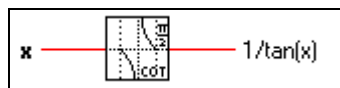
Cosinus

Calcule le cosinus de x , où x est en radians.



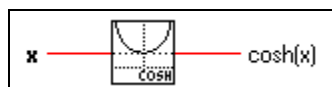
Cotangente

Calcule la cotangente de x , où x est en radians. La cotangente est l'inverse de la tangente.



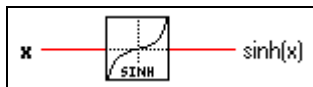
Cosinus hyperbolique

Calcule le cosinus hyperbolique de x .



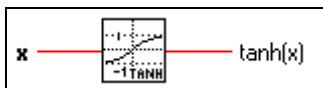
Sinus hyperbolique

Calcule le sinus hyperbolique de x .



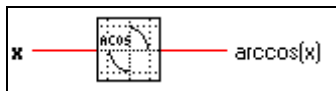
Tangente hyperbolique

Calcule la tangente hyperbolique de x .



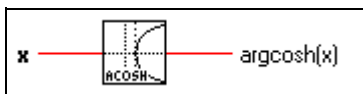
Cosinus inverse

Calcule l'arccosinus de x en radians. Si x n'est pas complexe et qu'il est inférieur à -1 ou supérieur à 1 , le résultat est NaN.



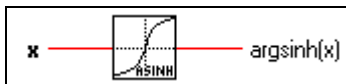
Cosinus inverse hyperbolique

Calcule l'arccosinus hyperbolique de x . Si x n'est pas complexe et qu'il est inférieur à 1 , le résultat est NaN.



Sinus inverse hyperbolique

Calcule l'arcsinus hyperbolique de x .



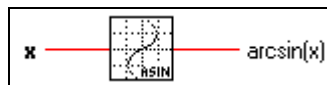
Tangente inverse hyperbolique

Calcule l'arctangente hyperbolique de x . Si x n'est pas complexe et qu'il est inférieur à -1 ou supérieur à 1 , le résultat est NaN.



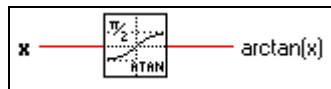
Inverse sinus

Calcule l'arcsinus de x en radians. Si x n'est pas complexe et qu'il est inférieur à -1 ou supérieur à 1 , le résultat est NaN.



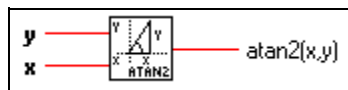
Tangente inverse

Calcule l'arctangente de x en radians (comprise entre $-\pi/2$ et $\pi/2$).



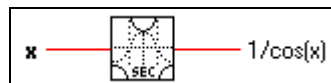
Tangente inverse (deux entrées)

Calcule l'arctangente de y/x en radians. Cette fonction peut calculer l'arctangente pour des angles contenus dans n'importe lequel des quatre quadrants du plan vectoriel x - y , tandis que la fonction Tangente inverse calcule l'arctangente dans deux quadrants uniquement.



Sécante

Calcule la sécante de x , où x est en radians.



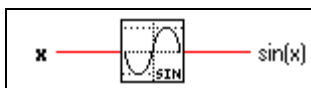
Sinc

Calcule le sinus de x divisé par x , où x est en radians.



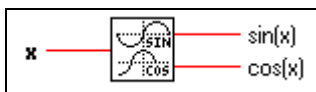
Sinus

Calcule le sinus de x , où x est en radians.



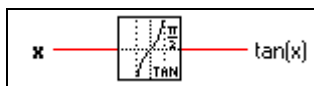
Sinus et cosinus

Calcule le sinus et le cosinus de x , où x est en radians. Utilisez cette fonction uniquement quand les deux résultats sont nécessaires.



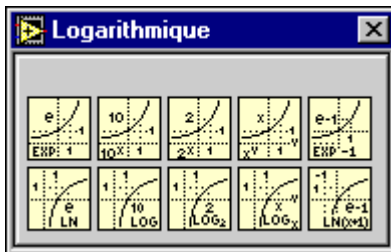
Tangente

Calcule la tangente de x , où x est en radians.



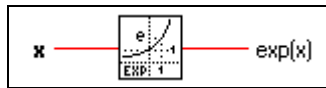
Descriptions des fonctions logarithmiques

L'illustration suivante montre les options de la sous-palette **Logarithmique**.



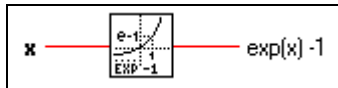
Exponentielle

Calcule la valeur de e à la puissance de x .



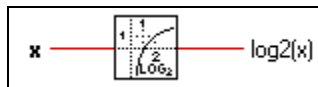
Exponentielle (Arg) -1

Calcule la valeur de e à la puissance x moins 1. Quand x est très petit, cette fonction est plus précise que d'exécuter la fonction Exponentiel et de soustraire 1 à la valeur obtenue.



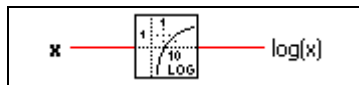
Logarithme de base 2

Calcule le logarithme de base 2 de x . Si x est égal à 0, $\log_2(x)$ est $-\infty$. Si x n'est pas complexe et inférieur à 0, $\log_2(x)$ est NaN.



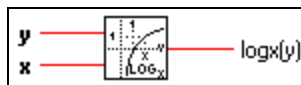
Logarithme de base 10

Calcule le logarithme de base 10 de x . Si x est égal à 0, $\log(x)$ est $-\infty$. Si x n'est pas complexe et inférieur à 0, $\log(x)$ est NaN.



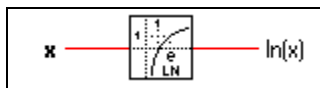
Logarithme de base X

Calcule le logarithme de base x de y ($x > 0$, $y > 0$). Si y est égal à 0, la sortie est $-\infty$. —Si x et x ne sont pas complexes et si x est inférieur ou égal à 0, ou si y est inférieur à 0, la sortie est NaN.



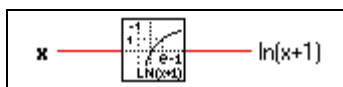
Logarithme naturel

Calcule le logarithme e de base naturelle de x . Si x est égal à 0, $\ln(x)$ est $-\infty$. Si x n'est pas complexe et inférieur à 0, $\ln(x)$ est NaN.



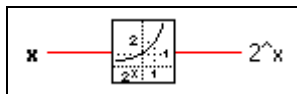
Logarithme naturel (Arg +1)

Calcule le logarithme naturel de $(x + 1)$. Quand x est proche de 0, cette fonction est plus précise que d'ajouter 1 à x et utiliser la fonction Logarithme naturel. Si x est égal à -1 , le résultat est $-\infty$. Si x n'est pas complexe et inférieur à -1 , le résultat est NaN.



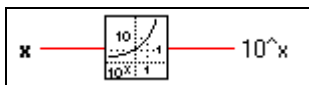
Puissance de 2

Calcule 2 à la puissance x .



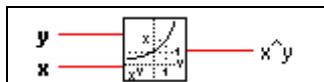
Puissance de 10

Calcule 10 à la puissance x .



Puissance de X

Calcule x à la puissance y . Si x n'est pas complexe, il doit être supérieur à zéro sauf si y est un entier. Autrement, le résultat est NaN. Si y est égal à zéro, x^y est égal à 1 pour toutes les valeurs de x , y compris zéro.



Descriptions des fonctions complexes

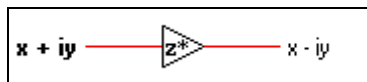
L'illustration suivante montre les options disponibles dans la sous-palette **Complexe**.



Les fonctions “Polaire en complexe” et “Re/Im en complexe” créent des nombres complexes à partir de deux valeurs en notation cartésienne ou polaire. Les fonctions “Complexe en polaire” et “Complexe en Re/Im” séparent un nombre complexe en composantes cartésiennes ou polaires.

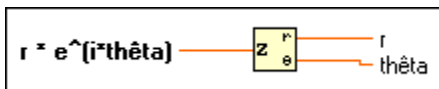
Complexe conjugué

Produit le complexe conjugué de $x + iy$.



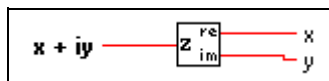
Complexe en polaire

Sépare un nombre complexe en composantes polaires.



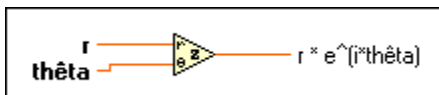
Complexe en Re/Im

Sépare un nombre complexe en composantes cartésiennes.



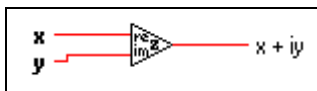
Polaire en complexe

Construit un nombre complexe à partir de deux valeurs en notation polaire.



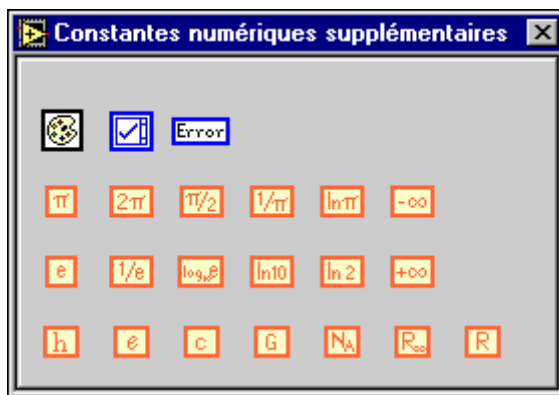
Re/Im en complexe

Construit un nombre complexe à partir de deux valeurs en notation cartésienne.



Descriptions des constantes numériques supplémentaires

L'illustration suivante montre les options disponibles dans la sous-palette **Constantes numériques supplémentaires**.



Les constantes supplémentaires définies par l'utilisateur

Vous pouvez définir les constantes suivantes.



Constante Symboles

Cette constante permet d'affecter des symboles à des éléments d'une commande de type menu déroulant. En général, vous devez relier cette constante à l'attribut node Item Symbols.



Constante Boîte de couleurs

Utilisez cette constante pour fournir une valeur de couleur constante au diagramme. Définissez cette valeur en cliquant sur la constante avec l'outil Doigt et en sélectionnant la couleur souhaitée.

La valeur de la constante boîte de couleurs ne peut pas être modifiée pendant l'exécution du VI. Vous pouvez affecter une étiquette à cette constante.

Error Constante Liste déroulante d'erreurs

Cette constante représente une liste déroulante prédéfinie d'erreurs spécifiques à l'utilisation de la mémoire, à la mise en réseau, à l'impression et aux E/S sur fichier. Les erreurs relatives à l'acquisition de données, au GPIB, à VISA, aux VIs Série et aux fonctions ne sont pas disponibles dans cette liste déroulante.

Constantes fixes

Les constantes suivantes sont fixes.

1/y Nombre d'Avogadro (1/mol)

Retourne la valeur 6,0220e23.

log_e Logarithme de base 10 de e

Retourne la valeur 0,43429448190325183.

c Charge élémentaire (c)

Retourne la valeur 1,6021892e-19.

G Constante gravitationnelle (Nm²/kg²)

Retourne la valeur 6,6720e-11.

R Constante des gaz parfaits (J/mol K)

Retourne la valeur 8,31441.

e

Retourne la valeur 2,7182818284590452e+0.

ln π Logarithme naturel de Pi

Retourne la valeur 1,14472988584940020.

ln 2 Logarithme naturel de 2

Retourne la valeur 0,69314718055994531.

ln 10 Logarithme naturel de 10

Retourne la valeur 2,30234095236904570.



Moins l'infini

Retourne la valeur $-\infty$.



Pi

Retourne la valeur 3,14159265358979320.



Pi divisé par 2

Retourne la valeur 1,57079632679489660.



Pi multiplié par 2

Retourne la valeur 6,28318530717958650.



Constante de Planck (J/Hz)

Retourne la valeur 6,6262e-34.



Plus l'infini

Retourne la valeur $+\infty$.



Inverse de e

Retourne la valeur 0,36787944117144232.



Inverse de Pi

Retourne la valeur 0,31830988618379067.



Constante de Rydberg (/m)

Retourne la valeur 1,097373177e7.



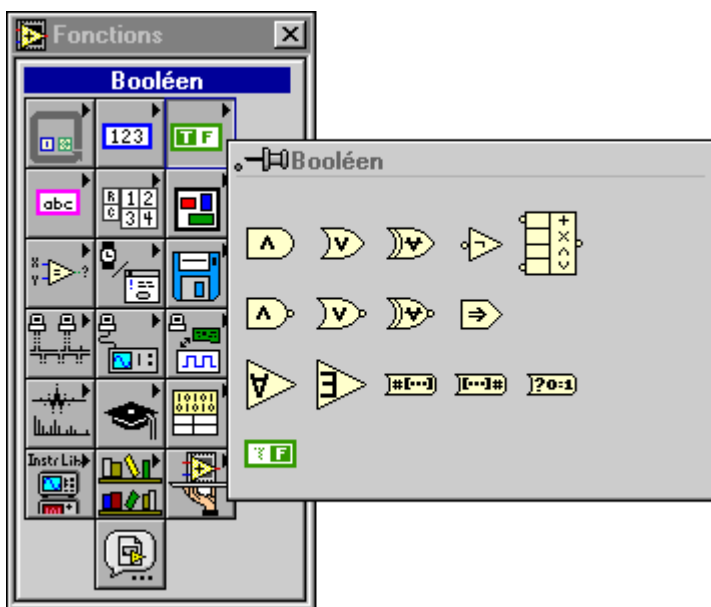
Vitesse de la lumière dans le vide (m/sec)

Retourne la valeur 299.792.458.

Fonctions booléennes

Ce chapitre décrit les fonctions utilisées pour effectuer des opérations logiques.

L'illustration suivante montre la palette **Booléen**, à laquelle vous pouvez accéder en sélectionnant **Fonctions**»**Booléen**.



Pour obtenir des exemples de quelques fonctions booléennes, reportez-vous à `examples\general\structs.llb`.

Polymorphisme des fonctions booléennes

Les fonctions logiques acceptent des données d'entrées booléennes ou numériques. Si l'entrée est numérique, le G exécute une opération en représentation binaire. Si l'entrée est un entier, la sortie a la même représentation. Si l'entrée est un nombre à virgule flottante, le G l'arrondit à un entier long et la sortie est un entier long.

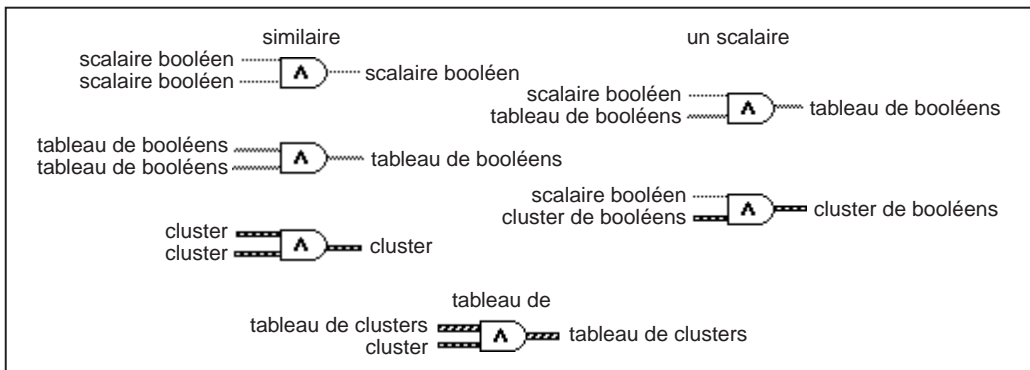
Les fonctions logiques agissent sur des tableaux de nombres ou de valeurs booléennes, des clusters de nombres ou de valeurs booléennes, des tableaux de clusters de nombres ou de valeurs booléennes, et ainsi de suite.

Une définition formelle et récursive du type d'entrée possible figure ci-dessous :

Type logique = scalaire booléen || scalaire numérique || tableau [types logiques] || cluster [types logiques]

Les nombres complexes et les tableaux de tableaux ne sont pas autorisés.

Les fonctions logiques à deux entrées peuvent avoir les mêmes combinaisons d'entrées que les fonctions arithmétiques. Néanmoins, les fonctions logiques comportent une restriction supplémentaire car les opérations de base ne peuvent avoir lieu qu'entre deux valeurs booléennes ou deux nombres. Par exemple, vous ne pouvez pas calculer un ET entre une valeur booléenne et un nombre. L'exemple ci-dessous illustre quelques combinaisons de valeurs booléennes possibles pour la fonction ET.

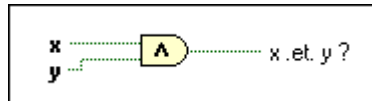



Descriptions des fonctions booléennes

Les fonctions booléennes suivantes sont disponibles.

ET

Calcule le ET logique des entrées.



 **Remarque** *Cette fonction exécute des opérations en représentation binaire sur des entrées numériques.*

ET logique entre les éléments du tableau

Retourne Vrai (TRUE) si tous les éléments du **Tableau de booléens** sont vrais ; sinon retourne Faux (FALSE).

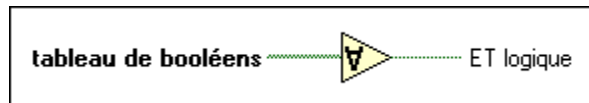


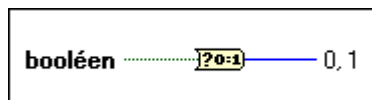
Tableau de booléens en nombre

Convertit **Tableau de booléens** en entier long non signé en l'interprétant comme une représentation en complément à deux, où l'élément 0 du tableau est le bit de moindre poids.



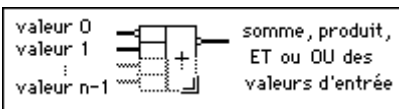
Booléen en (0,1)

Convertit une valeur booléenne en entier de type mot : 0 et 1 pour les valeurs d'entrée Faux (FALSE) et Vrai (TRUE), respectivement.




Opérateur arithmétique

Effectuent des opérations arithmétiques sur plusieurs entrées de type numérique, de cluster ou booléen.



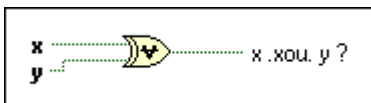
Vous pouvez choisir l'opération (multiplier, ET ou OU) en ouvrant le menu local de la fonction et en sélectionnant **Changer le mode**.

Vous pouvez inverser les entrées ou la sortie de cette fonction en ouvrant le menu local des terminaux séparément et en sélectionnant **Inverser**. Pour l'opération Ajouter, sélectionnez **Inverser** pour changer le signe d'une entrée ou de la sortie. Pour l'opération Multiplier, sélectionnez **Inverser** pour inverser une entrée ou la sortie. Pour l'opération ET ou OU, sélectionnez **Inverser** pour inverser logiquement une entrée ou la sortie.

 **Remarque** *Vous pouvez ajouter des entrées à ce nœud en ouvrant le menu local d'une entrée et en sélectionnant Ajouter une entrée ou en positionnant l'outil Flèche sur le coin inférieur gauche ou droit et en le faisant glisser.*

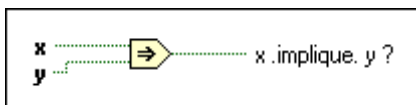
OU exclusif

Calcule le OU exclusif des entrées.



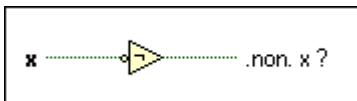
Implique

Calcule le OU logique de y et de la négation logique de x . La fonction inverse x et calcule ensuite le OU logique de y et de x inversé.



Non

Calcule la négation logique de l'entrée.



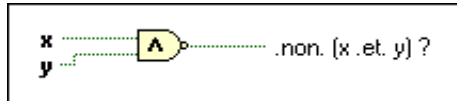
Non ET

Calcule le NON ET logique des entrées.



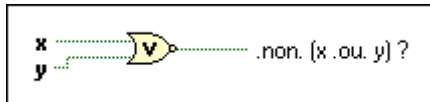
Non OU exclusif

Calcule la négation logique du OU logique exclusif des entrées.



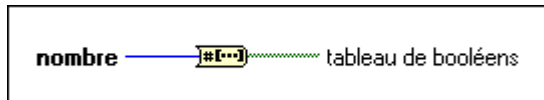
Non OU

Calcule le NON OU logique des entrées.



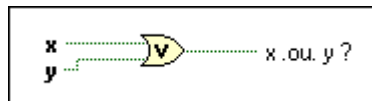
Nombre en tableau de booléens

Convertit le **nombre** en tableau de booléens de 8, 16 ou 32 éléments, où l'élément 0 correspond au bit de moindre poids de la représentation complémentaire à deux de l'entier.



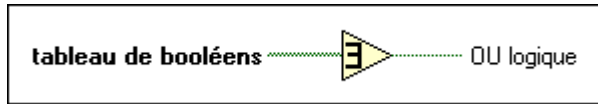
OU

Calcule le OU logique des entrées.



OU logique entre les éléments du tableau

Retourne Faux (FALSE) si tous les éléments du **Tableau de booléens** sont faux ; sinon retourne Vrai (TRUE) .



Constante booléenne

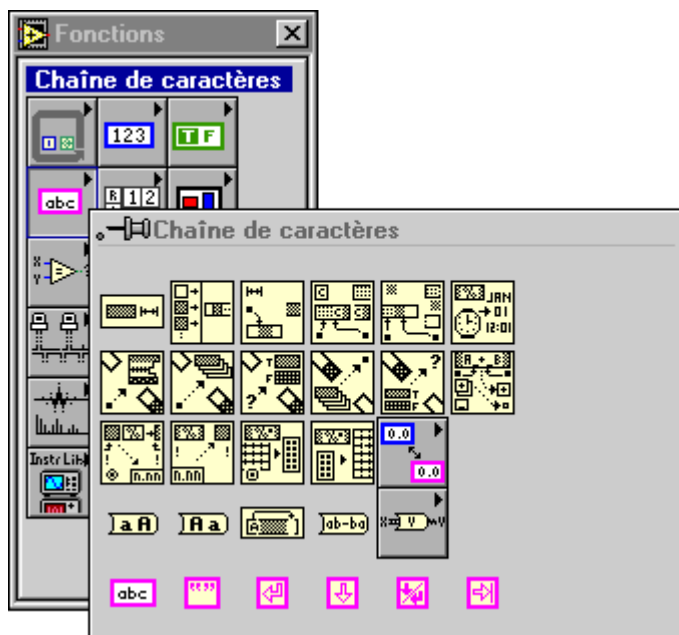
Utilisez cette constante pour fournir une valeur constante Vrai ou Faux (TRUE/FALSE) au diagramme. Définissez cette valeur en cliquant sur la partie **T** ou **F** de la constante avec l'outil Doigt. Cette valeur ne peut pas être modifiée pendant l'exécution du VI. Vous pouvez affecter une étiquette à cette constante.



Fonctions de chaînes de caractères

Ce chapitre décrit les fonctions de chaînes de caractères, notamment celles qui convertissent les chaînes de caractères en nombres et les nombres en chaînes de caractères.

L'illustration suivante montre la palette **Chaîne de caractères**, à laquelle vous pouvez accéder en sélectionnant **Fonctions»Chaîne de caractères**.



Vue d'ensemble du polymorphisme des fonctions de chaînes de caractères

Cette section présente une description du polymorphisme des fonctions de chaînes de caractères, les fonctions Chaîne vers Nombre supplémentaires et les fonctions de conversion de chaînes de caractères.

Polymorphisme des fonctions de chaînes de caractères

Les fonctions “Longueur de chaîne de caractères”, “En majuscules”, “En minuscules”, “Renverser une chaîne de caractères” et “Rotation dans une chaîne de caractères” acceptent des chaînes de caractères, des clusters, des tableaux de chaînes de caractères et des tableaux de clusters. “En majuscules” et “En minuscules” acceptent également des nombres, des clusters de nombres et des tableaux de nombres, en les interprétant comme des caractères exprimés en code ASCII (consultez l'Annexe C, [Messages d'interface multiligne GPIB](#), pour obtenir la liste de codes correspondant à chaque caractère). Les entrées de largeur et de précision doivent être scalaires.

Polymorphisme des fonctions de conversion supplémentaire

Les fonctions “Au format décimal”, “Au format hexadécimal”, “Au format octal”, “Au format ingénieur”, “Au format fractionnel” et “Au format exponentiel” acceptent des clusters et des tableaux de nombres et produisent des clusters et des tableaux de chaînes de caractères. “A partir d'un format décimal”, “A partir d'un format hexa”, “A partir d'un format octal” et “A partir d'un format exp/fract/sci” acceptent des clusters et des tableaux de chaînes de caractères et produisent des clusters et des tableaux de nombres. Les entrées de largeur et de précision doivent être scalaires.

Polymorphisme des fonctions de conversion de chaînes de caractères

Les fonctions “Chemin en une chaîne de caractères” et “Chaîne de caractères en chemin” sont polymorphes. Elles agissent sur des valeurs scalaires, des tableaux de scalaires, des clusters de scalaires, des tableaux de clusters de scalaires, et ainsi de suite. La sortie a la même composition que l'entrée mais son type est différent.

Vue d'ensemble des chaînes de format

Plusieurs fonctions G disposent d'une entrée **chaîne de format**, qui contrôle le comportement de la fonction. Une chaîne de format comporte une ou plusieurs spécifications de format, qui déterminent quelle action prendre pour traiter un paramètre donné. Les fonctions "Formater en une chaîne de caractères" et "Balayer une chaîne de caractères" peuvent utiliser plusieurs spécifications de format dans la chaîne de format, une pour chaque entrée ou sortie redimensionnable de la fonction. Les caractères de la chaîne qui ne font pas partie de la spécification de format sont copiés directement dans la chaîne de caractères de sortie (dans le cas de "Formater en une chaîne de caractères") ou sont comparés à la chaîne d'entrée (dans le cas de "Balayer une chaîne de caractères"), excepté pour des codes spéciaux d'échappement. Vous pouvez utiliser ces codes pour insérer des caractères non affichables, des barres obliques inverses et des caractères de pourcentage dans n'importe quelle chaîne de format. Ces codes sont comparables aux codes du langage de programmation C.

Le tableau 6-1 affiche les codes spéciaux d'échappement. Aucun code n'existe pour le caractère fin de la ligne dépendant de la plate-forme. Si vous devez en ajouter un, utilisez la constante "Fin de ligne" dans la palette **Chaîne de caractères**.

Tableau 6-1. Codes spéciaux d'échappement

Code	Signification
\r	Retour chariot
\t	Tab
\b	Retour arrière
\n	Nouvelle ligne
\f	Saut de page
\s	Espace
\xx	Caractère avec code ASCII hexadécimal xx (en utilisant 0 à 9 et majuscule A à F)
\\	\
%%	%

Observez que pour les fonctions “Balayer une chaîne de caractères” et “Formater & Dépouiller”, un espace dans la chaîne de format correspond à n’importe quel espace blanc (espaces, tabulations et sauts de page) dans la chaîne d’entrée.

Les fonctions “Formater & Ajouter”, “Formater & Dépouiller”, “Tableau en une chaîne de caractères format tableur” et “Chaîne de caractères au format tableur en tableau” utilisent uniquement une spécification de format dans la chaîne de format car ces fonctions n’ont qu’une entrée à convertir. Toute spécification extérieure insérée dans ces fonctions est perçue comme une chaîne de caractères littérale sans signification spéciale.

Pour les fonctions qui produisent une chaîne de caractères comme sortie, (“Formater en une chaîne de caractères”, “Formater & Ajouter” et “Tableau en une chaîne de caractères format tableur” entre autres), la spécification de format respecte la syntaxe suivante. Deux crochets ([]) entourent les éléments en option.

`%[-][+][^][0][Largeur][.Précision][{unité}]`

Code de conversion

Pour les fonctions qui balayent une chaîne de caractères, comme “Balayer une chaîne de caractères”, “Formater & Dépouiller” et “Chaîne de caractères au format tableur en tableau”, la spécification de format respecte la syntaxe simplifiée suivante :

`%[Longueur]Code de conversion`

Le tableau 6-2 affiche les syntaxes de chaînes de caractères disponibles.

Tableau 6-2. Syntaxe de chaînes de caractères

Élément de syntaxe	Description
%	Commence la spécification de format.
- (optionnel)	Fait que le paramètre est justifié à gauche au lieu d’être justifié à droite dans le sens de sa largeur.
+ (optionnel)	Pour les paramètres numériques, inclut le signe même si le nombre est positif.
^ (optionnel)	Lorsqu’utilisé avec les codes de conversion e ou g, utilise la notation ingénieur (l’exposant est toujours un multiple de 3).

Tableau 6-2. Syntaxe de chaînes de caractères (Suite)

Élément de syntaxe	Description
0 (optionnel)	Comble tout espace excédentaire à gauche d'un paramètre numérique avec des 0 et non avec des espaces.
Largeur (optionnel)	<p>Pendant le balayage, spécifie une largeur de champ exacte à utiliser. Le G ne balaye que le nombre de caractères spécifié lors du traitement du paramètre.</p> <p>Pendant le formatage, spécifie la largeur minimale du champ de caractères de la sortie. Cette largeur ne définit pas une largeur maximale ; le G utilise autant de caractères qu'il faut pour formater le paramètre sans le tronquer. Le G comble le champ à gauche ou à droite du paramètre avec des espaces, selon la justification. Si Largeur manque ou est égale à zéro, la sortie est juste assez longue pour pouvoir contenir le paramètre d'entrée.</p>
.	Sépare Largeur de Précision.
Précision (optionnel)	<p>Pour les paramètres à virgule flottante, spécifie le nombre de chiffres à droite de la virgule décimale. Si Largeur n'est pas suivie d'un point, le G insère une partie fractionnelle de six chiffres. Si Largeur est suivie d'un point et Précision manque ou est égale à 0, le G n'insère pas de partie fractionnelle.</p> <p>Pour les paramètres de chaîne de caractères, spécifie la largeur maximale du champ. Le G tronque les chaînes de caractères qui dépassent cette longueur.</p>
{unité} (optionnel)	Supplante le choix d'unité d'un VI pendant la conversion en une quantité physique (une valeur avec une unité associée). Doit être une unité valable.

Tableau 6-2. Syntaxe de chaînes de caractères (Suite)

Élément de syntaxe	Description
Codes de conversion	Caractère simple spécifiant comment balayer ou formater un périmètre, comme suit : d entier décimal x entier hexa o entier octal b entier binaire f nombre à virgule flottante à format fractionnel e nombre à virgule flottante à notation scientifique g nombre à virgule flottante utilisant le format e si l'exponentiel est inférieur à -4 ou plus grand que Précision, ou le format f dans le cas contraire s Chaîne de caractères
Codes de localisation	Codes utilisés comme séparateurs de format pour la localisation, comme suit : % , ; séparateur à virgule décimale % . ; séparateur à point décimal % ; séparateur par défaut du système

Les codes de conversion utilisés en G sont comparables à ceux du langage de programmation C. Cependant, le G utilise des codes de conversion pour déterminer le format textuel du paramètre et non son type de données.

Vous pouvez utiliser les codes de conversion d, x, o, b, f, e et g pour traiter n'importe quel type de données numériques en G, notamment des énums et des nombres complexes.

Pour les nombres complexes, vous pouvez utiliser la spécification de format pour traiter les parties réelles et imaginaires comme des paramètres uniques.

Vous pouvez utiliser le code de conversion s pour traiter des paramètres de chaîne de caractères ou de chemins, ou des énums.

Notez que vous pouvez utiliser un code de conversion numérique ou de chaîne de caractères avec une énum, selon que vous souhaitez obtenir une valeur numérique ou symbolique (chaîne de caractères) d'énum.

Pour assurer la compatibilité avec le C, le G traite un code de conversion u (entier non signé) de la même façon qu'un d et ignore un l ou L qui précède le code de conversion. Néanmoins, en G, le type de données du paramètre détermine la taille d'un entier et si l'entier est signé ou non signé.

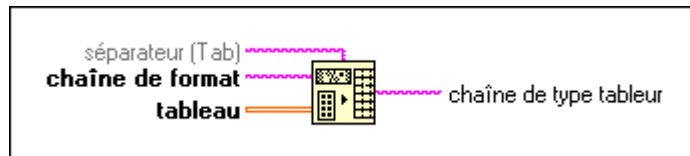
Pour des exemples de chaînes de format, reportez-vous aux descriptions des fonctions "Formater en une chaîne de caractères" et "Balayer une chaîne de caractères" figurant plus loin dans ce chapitre.

Descriptions des fonctions de chaînes de caractères

Les fonctions de chaînes de caractères suivantes sont disponibles.

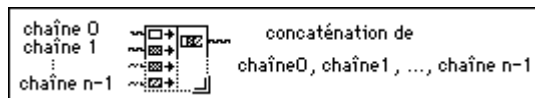
Tableau en une chaîne de caractères au format tableur

Convertit un **tableau** de n'importe quelle dimension en une chaîne **de type tableur**. La **chaîne de type tableur** constitue un tableau présenté sous forme de chaîne de caractères, qui comporte des éléments de colonne séparés par des séparateurs, des rangées séparées par des caractères de fin de ligne qui dépendent de la plate-forme et, pour les tableaux à trois dimensions ou plus, des pages séparées.



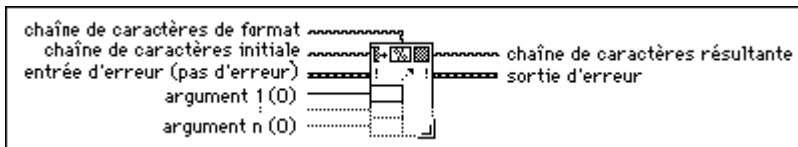
Concaténer des chaînes de caractères

Assemble des chaînes de caractères et des tableaux unidimensionnels des chaînes de caractères d'entrée en une chaîne de caractères simple de sortie. Si les entrées sont des tableaux, cette fonction assemble chaque élément du tableau.



Formater en une chaîne de caractères

Convertit des arguments d'entrée en une **chaîne de caractères résultante** dont le format est déterminé par la **chaîne de format**. Vous pouvez augmenter le nombre de paramètres en ouvrant le menu local du nœud et en sélectionnant **Ajouter un paramètre** ou en plaçant l'outil Flèche sur le coin inférieur gauche ou droit du nœud et en l'étirant jusqu'à ce que vous atteigniez le nombre souhaité d'arguments.



Le tableau 6-3 montre les erreurs qui peuvent apparaître dans **Sortie d'erreur** avec la fonction "Formater en une chaîne de caractères".

Tableau 6-3. Erreurs possibles de la fonction "Formater en une chaîne de caractères"

Erreur	Code	Description
Mauvaise correspondance du type de spécification de format	81	Le type de données de la spécification de format contenue dans la chaîne de format ne correspond pas au type de données de l'argument d'entrée respectif.
Spécification de format inconnue	82	La chaîne de format comporte une spécification de format non valable.
Pas assez de spécifications de format	83	Il y a plus d'arguments que de spécifications de format.
Trop de spécifications de format	84	Il y a plus de spécifications de format que d'arguments.



Remarque *Si une erreur survient, la composante source du cluster d'erreur de sortie affiche une chaîne de caractères de la forme "Scan from String (arg n)", où n représente le premier argument pour lequel l'erreur est survenue.*

Si vous reliez une constante de chaîne de caractères à l'entrée **chaîne de format**, le G vérifie s'il y a des erreurs dans la **chaîne de format** lors de la compilation. Vous devez corriger de telles erreurs avant d'exécuter le VI. Dans ce cas, aucune erreur ne peut survenir pendant l'exécution.

Exemples de spécifications de format

Dans le tableau 6-4, le caractère de soulignement (_) représente des espaces dans la chaîne de la sortie. Les trois derniers éléments sont des exemples d'entrées de quantités physiques.

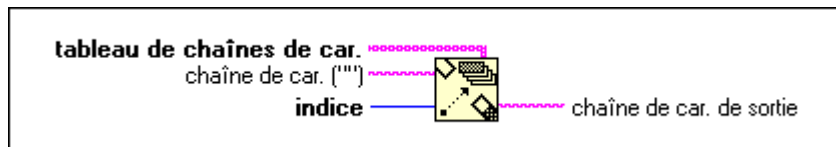
Tableau 6-4. Spécifications de format

Chaîne de format	Argument(s)	Chaîne de caractères résultante
cote = %2d%%	87	cote = 87%
niveau = \n%-7,2e V	0,03642	niveau = 3,64e-2 V
nom: %s, %s.	Dupond Henri	nom: Dupond, Henri.
Temp: %05,1f %s	96,793 Fahrenheit	Temp: 096,8 Fahrenheit
Chaîne de caractères: %10,5s.	Bonjour, tous	Chaîne de caractères: _____Bonjour.
%5,3f	5,67 N	5,670 N
%5,3{mN}f	5,67 N	5670,000 mN
%5,3{kg}f	5,67 N	5,670 ?kg

La dernière entrée montre la sortie obtenue quand l'unité de la spécification de format est en conflit avec l'unité d'entrée.

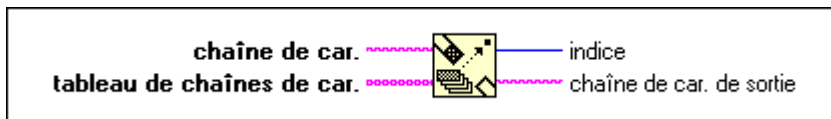
Indexer & Ajouter

Sélectionne une chaîne de caractères spécifiée par l'indice d'un **tableau de chaîne de caractères** et l'ajoute à une **chaîne de caractères**.



Indexer & Dépouiller

Compare chaque chaîne de caractères d'un **tableau de chaîne de caractères** avec le début de **chaîne de caractères** jusqu'à trouver une correspondance.



Rechercher une expression

Cherche une **expression** dans une **chaîne de caractères** à partir de **offset**. Si une correspondance est obtenue, cette fonction divise la **chaîne de caractères** en trois sous-chaînes de caractères.

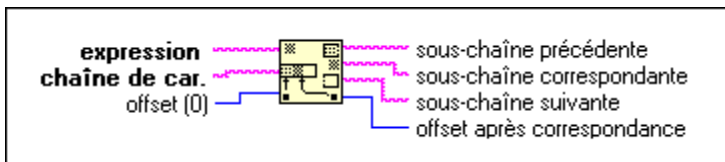


Tableau 6-5. Caractères spéciaux pour la recherche d'expressions

Caractère spécial	Interprétation de la fonction “Rechercher une expression comme...”
.	Correspond à n'importe quel caractère.
?	Correspond à une ou zéro instance de l'expression qui précède ?.
\	Annule l'interprétation de caractères spéciaux (par exemple, \? correspond au point d'interrogation). Vous pouvez également utiliser les constructions suivantes pour les espaces et les caractères non affichables : \b retour arrière \f saut de page \n nouvelle ligne \s espace \r retour chariot \xxx n'importe quel caractère, où xx représente le code hexa utilisant les chiffres 0 à 9 et les majuscules A à F \t tabulation

Tableau 6-5. Caractères spéciaux pour la recherche d'expressions (Suite)

Caractère spécial	Interprétation de la fonction “Rechercher une expression comme...”
^	Si ^ est le premier caractère de expression , elle ancre la correspondance à l' offset dans la chaîne de caractères . La correspondance échoue sauf si expression correspond à la portion de chaîne de caractères qui commence par le caractère situé à l' offset . Si ^ n'est pas le premier caractère, il est interprété comme un caractère courant.
[]	Entoure des options facultatives. Par exemple [abc] correspond à a, b, ou c. Le caractère ci-après a une signification spéciale lorsqu'utilisé entre crochets : – (tiret) Indique une gamme lorsqu'utilisé entre deux chiffres, deux lettres minuscules ou majuscules (par exemple, [0–5],[a–g], ou [L–Q]) Les caractères ci-après n'ont une signification que lorsqu'ils représentent le premier caractère entre crochets : ~Exclut un ensemble de caractères, dont des caractères non affichables. [~0–9] correspond à n'importe quel caractère autre que 0 à 9. ^Exclut un ensemble de caractères et s'applique à tous les caractères affichables (et les caractères espace). [^0–9] correspond au caractère espace et tous les caractères affichables excepté 0 à 9.
+	Correspond au nombre le plus grand d'instances de l'expression précédent + ; il faut au moins une instance pour obtenir une correspondance.
*	Correspond au nombre le plus grand d'instances de l'expression précédent * dans expression , y compris zéro instance.
\$	Si \$ est le dernier caractère expression , il ancre la correspondance au niveau du dernier élément de chaîne de caractères . La correspondance échoue à moins que expression ne corresponde jusqu'au dernier caractère de la chaîne (dernier caractère compris). Si \$ n'est pas en dernier, il est interprété comme un caractère courant.

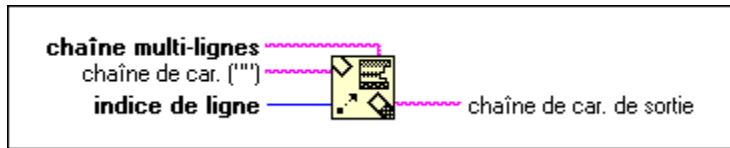
Le tableau 6-6 présente des exemples de chaînes de caractères pour les fonctions de recherche d'expressions.

Tableau 6-6. Exemples de chaînes de caractères pour la recherche d'expressions

Caractères à faire correspondre	Expression
VOLTS	VOLTS
Toutes les versions en majuscules et en minuscules de volts, c'est-à-dire, VOLTS, Volts, volts, et ainsi de suite	[Vv][Oo][Ll][Tt][Ss]
Un espace, un signe d'addition ou de soustraction	[+-]
Une séquence d'un ou de plusieurs chiffres	[0-9]+
Zéro ou plusieurs espaces	\s* ou * (c'est-à-dire, un espace suivi d'un astérisque)
Un ou plusieurs espaces, tabs, nouvelles lignes ou retours chariot	[\t \r \n \s]+
Un caractère ou plus autres que des chiffres	[~0-9]+
Le mot Niveau uniquement s'il commence à la position offset dans la chaîne de caractères	^Niveau
Le mot Volts uniquement s'il apparaît à la fin de la chaîne de caractères	Volts\$
La plus longue chaîne de caractères entre parenthèses	(.*)
La plus longue chaîne de caractères entre parenthèses mais qui ne comprend aucune autre parenthèse à l'intérieur	([~()]*)
Le caractère [[[]

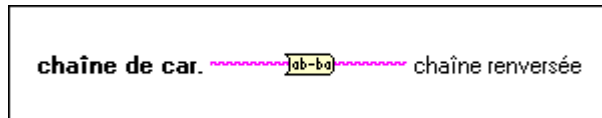
Piocher une ligne & Ajouter

Sélectionne une ligne d'une **chaîne de caractères multiligne** et ajoute cette ligne à une **chaîne de caractères**.



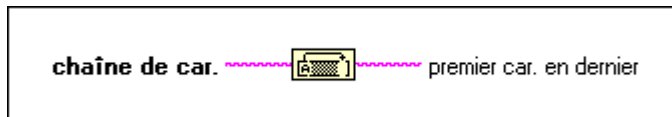
Renverser une chaîne de caractères

Produit une chaîne de caractères dont les caractères sont dans l'ordre inverse à celui de la **chaîne de caractères**.



Rotation dans une chaîne de caractères

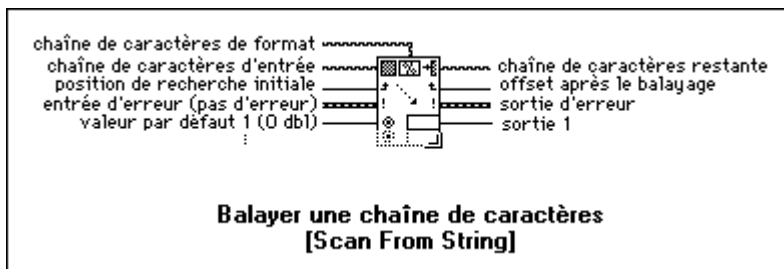
Place le premier caractère de **chaîne de caractères** à la fin de la chaîne **premier car. en dernier**, et décale les autres caractères d'un cran. La chaîne de caractères *abcd* devient par exemple *bcd a*.



Balayer une chaîne de caractères

Balaye la chaîne d'entrée et la convertit selon la **chaîne de format**. Vous pouvez augmenter le nombre de paramètres en ouvrant le menu local du nœud et en sélectionnant **Ajouter un paramètre** ou en posant l'outil Flèche sur le coin inférieur gauche ou droit du nœud et en l'étirant jusqu'à ce que vous atteigniez le nombre souhaité de paramètres.

Utilisez “Balayer une chaîne de caractères” quand vous connaissez le format exact de la chaîne d’entrée.



Le tableau 6-7 liste les erreurs de la fonction “Balayer une chaîne de caractères”.

Tableau 6-7. Erreurs “Balayer une chaîne de caractères”

Erreur	Code	Description
Mauvaise correspondance du type de spécification de format	81	Le type de données de la spécification de format contenu dans la chaîne de format ne correspond pas au type de données de sortie respectif.
Spécification de format inconnue	82	La chaîne de format comporte une spécification de format non valable.
Pas assez de spécifications de format	83	Il y a plus d’arguments que de spécifications de format.
Trop de spécifications de format	84	Il y a plus de spécifications de format que d’arguments.
Echec de balayage	85	“Balayer une chaîne de caractères” n’a pas pu convertir la chaîne d’entrée en type de données indiqué par la spécification de format.



Remarque *Si une erreur survient, la composante source du cluster de la sortie d’erreur affiche une chaîne de caractères de la forme “Scan from String (arg n)”, où n représente le premier argument pour lequel l’erreur est survenue.*

Si vous reliez une constante de chaîne de caractères à une chaîne de format, le G vérifie s’il y a des erreurs dans la chaîne de format lors de la compilation. De telles erreurs doivent être corrigées avant d’exécuter un VI. Dans cette instance, seule l’erreur “Echec de balayage” peut avoir lieu pendant l’exécution.

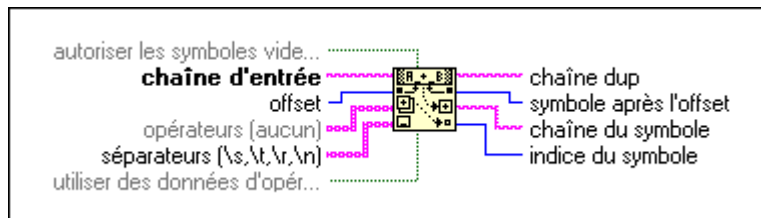
Le tableau 6-8 présente des exemples pour la fonction “Balayer une chaîne de caractères”.

Tableau 6-8. Exemples pour la fonction “Balayer une chaîne de caractères”

Chaîne d'entrée	Chaîne de format	Valeur(s) par défaut	Sortie(s)	Chaîne de caractères restante
abc xyz 12,3+56i 7200	%s %s%f%2d	— — 0&0i (CDB) —	abc xyz 12,3+56i 72	00
Q+1,27E-3 tail	Q%f t	—	1,27E-3	ail
0123456789	%3d%3d	—	12 345	6789
X:9.860 Z:3,450	X:%fY:%f	100 (I32) 100,0 (DBL)	10 100,0	Z: 3450
set49.4.2	set%d	—	49	.4.2

Rechercher un symbole

Balaye une **chaîne d'entrée**, à partir de **offset** et retourne le symbole trouvé suivant.

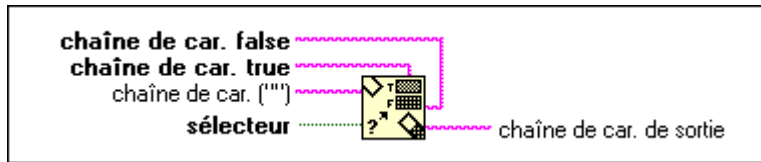


Un *symbole* est une sous-chaîne de caractères de la **chaîne d'entrée** entourée de **séparateurs** ou correspondant à un élément de **opérateurs**. En général, les symboles représentent des mots clés, des valeurs numériques ou des **opérateurs** individuels trouvés pendant l'analyse d'un fichier de configuration ou de tout autres données basées sur du texte. Cette fonction balaye la chaîne d'entrée, à partir de offset et retourne le symbole trouvé suivant.

Reportez-vous à la documentation en ligne pour plus d'informations sur la fonction “Rechercher un symbole” et ses paramètres.

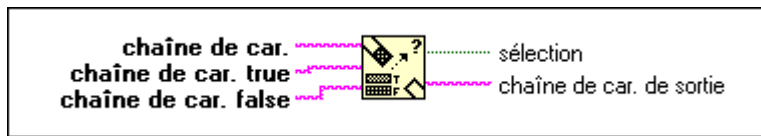
Sélectionner & Ajouter

Sélectionne **chaîne de car. false** ou **chaîne de car. true** par le biais d'un **sélecteur** booléen et l'ajoute à **Chaîne de caractères**.



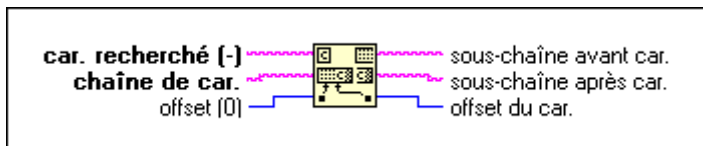
Sélectionner & Dépouiller

Examine le début de **chaîne de caractères** pour vérifier si elle correspond à **chaîne de car. false** ou **chaîne de car. true**. Cette fonction retourne une valeur booléenne Vrai (TRUE) ou Faux (FALSE) dans **sélection** selon que **chaîne de caractères** correspond à **chaîne de car. fausse (false)** ou **chaîne de car. vraie (true)**.



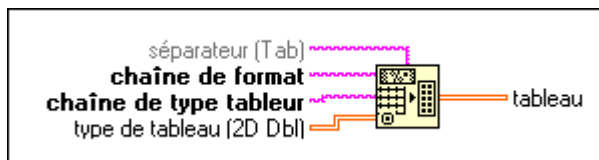
Eclater une chaîne de caractères

Coupe la chaîne de caractères au niveau de l'offset ou recherche la première instance de **car. recherché** dans **chaîne de caractères** à partir de **offset** et coupe la chaîne de caractères à cet endroit.



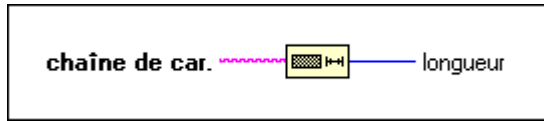
Chaîne de caractères au format tableau en tableau

Convertit une **chaîne de type tableau** en un **tableau** numérique dont la dimension et la représentation sont définies par l'entrée **type de tableau**. Cette fonction fonctionne pour les tableaux de chaînes de caractères ainsi que pour les tableaux de nombres.



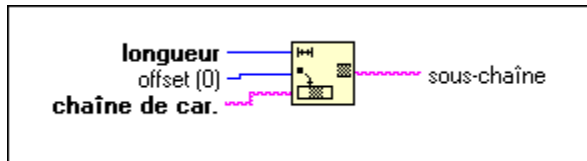
Longueur d'une chaîne de caractères

Retourne le nombre de caractères (octets) contenus dans **chaîne de caractères**.



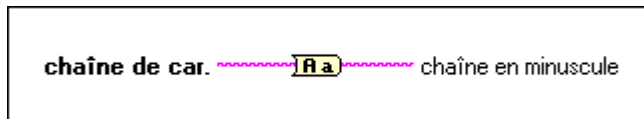
Sous-ensemble d'une chaîne de caractères

Retourne une **sous-chaîne** de la **chaîne de caractères** d'origine, commençant à **offset** et comprenant le nombre de caractères définis par **longueur**.



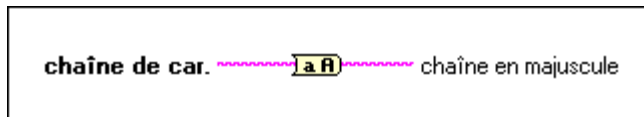
En minuscules

Convertit tous les caractères alphabétiques de la **chaîne de caractères** en minuscules. Cette fonction n'agit pas sur les caractères non alphabétiques.



En majuscules

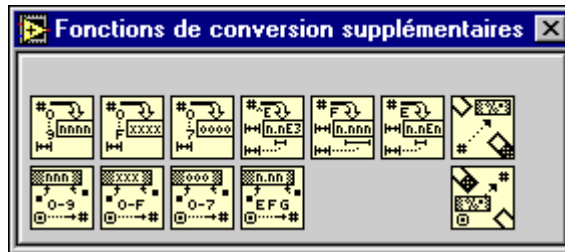
Convertit tous les caractères alphabétiques de la **chaîne de caractères** en majuscules. Cette fonction n'agit pas sur les caractères non alphabétiques.



Descriptions des fonctions de conversion supplémentaires

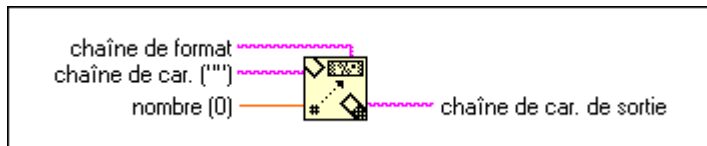
Pour des informations générales sur les fonctions de conversion supplémentaires, reportez-vous à la section *Polymorphisme pour des fonctions de conversion supplémentaires*, figurant plus haut dans ce chapitre.

L'illustration suivante montre les options disponibles dans la sous-palette **Fonctions de conversion supplémentaires**.



Formater & Ajouter

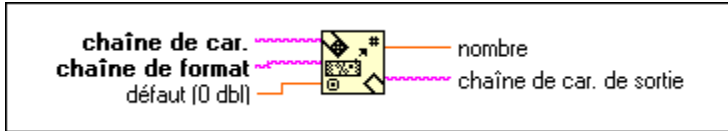
Convertit un **nombre** en une chaîne de caractères selon le format spécifié dans **chaîne de format** et l'ajoute à **chaîne de caractères**.



Remarque *La fonction “Formater en une chaîne de caractères” a la même fonctionnalité que “Formater & Ajouter”. Elle peut cependant utiliser plusieurs entrées et permet ainsi de convertir des informations simultanément. Pensez à utiliser la fonction “Formater en une chaîne de caractères” au lieu de cette fonction pour simplifier votre diagramme.*

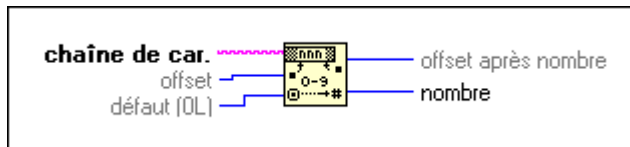
Formater & Dépouiller

Cherche une **chaîne de format** au début de **chaîne de caractères**, effectue le formatage de n'importe quel nombre trouvé dans cette partie de la chaîne de caractères selon les codes de conversion défini dans la **chaîne de format** ; retourne le nombre converti dans la sortie **nombre** et le reste de la **chaîne de caractères** après la correspondance dans la sortie **chaîne de car. de sortie**.



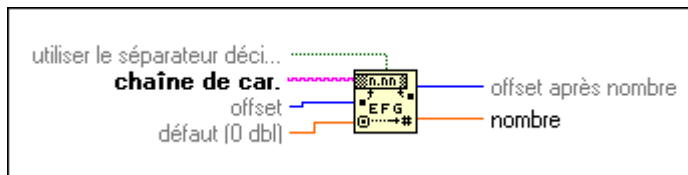
A partir d'un format décimal

Convertit les caractères numériques contenus dans la **chaîne de caractères**, à partir de **offset**, en entier décimal et le retourne dans la sortie **nombre**.



A partir d'un format exponentiel/fract/ing

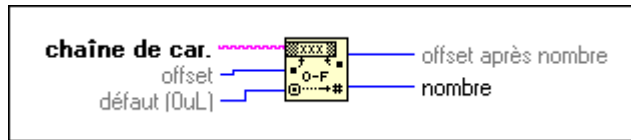
Interprète les caractères 0 à 9, addition, soustraction, e, E et la ponctuation décimale (c'est une virgule en général) contenus dans **chaîne de caractères**, à partir de **offset** comme un nombre à virgule flottante en notation ingénieur ou au format exponentiel ou fractionnel et le retourne dans la sortie **nombre**.



Remarque *Si vous reliez les caractères **Inf** ou **NaN** à l'entrée Chaîne de caractères, cette fonction retourne les valeurs en **G Inf** et **NaN**, respectivement.*

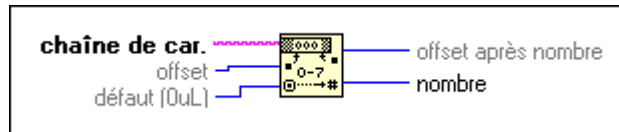
A partir d'un format hexa

Interprète les caractères 0 à 9, A à F, et a à f contenus dans **chaîne de caractères**, à partir de **offset** comme hexa et le retourne dans la sortie **nombre**.



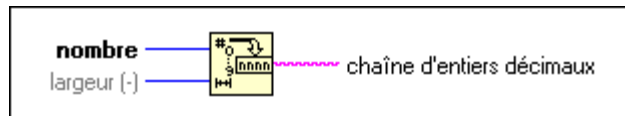
A partir d'un format octal

Interprète les caractères 0 à 7 contenus dans **chaîne de caractères**, à partir de **offset** comme entier octal et le retourne dans la sortie **nombre**. Cette fonction retourne également l'indice du premier caractère de la chaîne suivant le nombre.



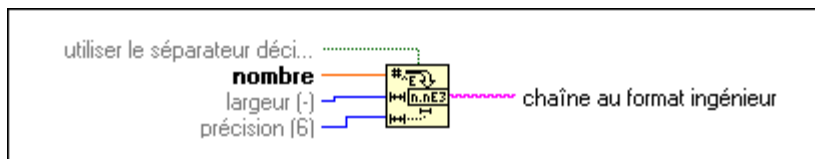
Au format décimal

Convertit un **nombre** en une chaîne de chiffres décimaux contenant **largeur** caractères ou plus si nécessaire.



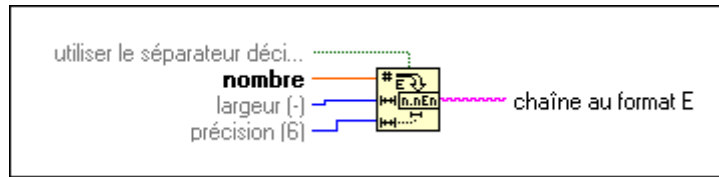
Au format ingénieur

Convertit un **nombre** en une chaîne de caractères à virgule flottante, au format ingénieur, contenant **largeur** caractères ou plus si nécessaire. Le format ingénieur ressemble au format E, excepté que l'exposant doit être un multiple de trois (-3, 0, 3, 6).



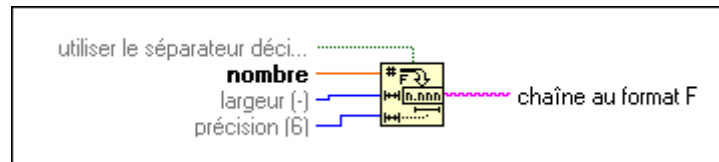
Au format exponentiel

Convertit un **nombre** en une chaîne de caractères à virgule flottante, au format E (notation exponentielle), contenant **largeur** caractères ou plus si nécessaire.



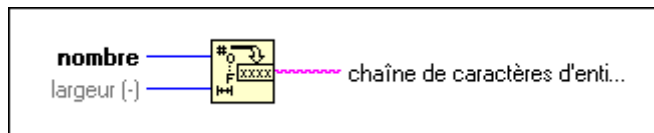
Au format fractionnel

Convertit un **nombre** en une chaîne de caractères à virgule flottante, au format F (notation fractionnelle), contenant **largeur** caractères ou plus si nécessaire.



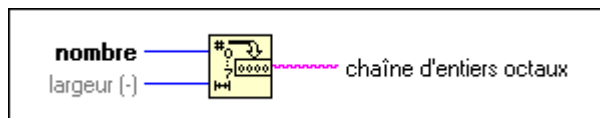
Au format hexadécimal

Convertit un **nombre** en une chaîne de chiffres hexadécimaux, contenant **largeur** caractères ou plus si nécessaire.



Au format octal

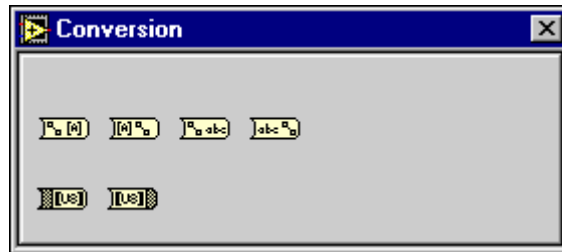
Convertit un **nombre** en une chaîne de chiffres octaux, contenant **largeur** caractères ou plus si nécessaire.



Descriptions des fonctions de conversion de chaîne de caractères

Pour des informations générales sur les fonctions de conversion de chaîne de caractères, reportez-vous à la section *Vue d'ensemble du polymorphisme des fonctions de chaînes de caractères*, figurant plus haut dans ce chapitre.

L'illustration suivante montre la sous-palette **Conversion**.



“Tableau de chaînes en chemin” accepte des tableaux unidimensionnels (1D) de chaînes de caractères, “Chemin en tableau de chaînes de caractères” accepte des chemins, “Chemin en une chaîne de caractères” accepte des chemins et “Chaîne de caractères en chemin” accepte des chaînes de caractères.

Tableau de chaînes en chemin

Convertit un **tableau de chaînes de caractères** en **chemin** relatif ou absolu.

Si le tableau contient une chaîne de caractères vide, l'emplacement du répertoire situé avant la chaîne de caractères vide est effacé dans la sortie de chemin. Pensez à cette modification comme à la progression à un niveau supérieur dans la hiérarchie des répertoires.

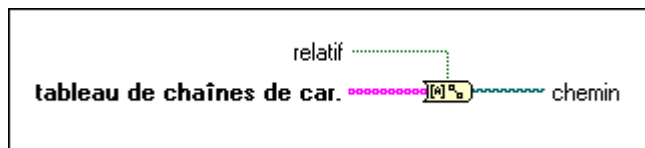


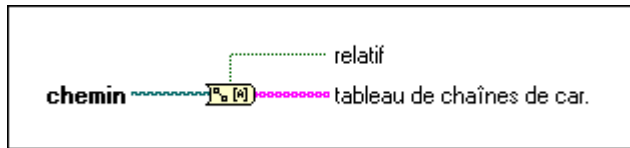
Tableau d'octets en une chaîne de caractères

Convertit un tableau d'octets non signés en une chaîne de caractères.



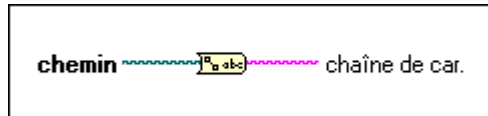
Chemin en tableau de chaînes de caractères

Convertit un **chemin** en un **tableau de chaînes de caractères** et indique si le chemin est **relatif**.



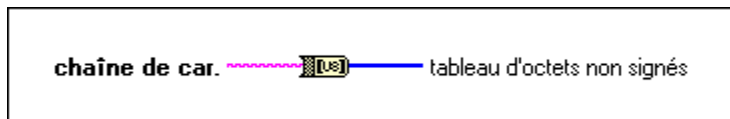
Chemin en une chaîne de caractères

Convertit un **chemin** en une chaîne de caractères décrivant un chemin au format standard de la plate-forme.



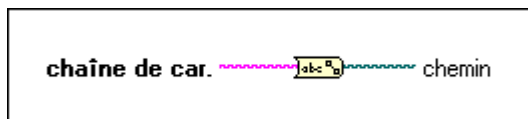
Chaîne de caractères en tableau d'octets

Convertit une **chaîne de caractères** en tableau d'octets non signés.



Chaîne de caractères en chemin

Convertit une chaîne de caractères, décrivant un chemin au format standard de la plate-forme courante, en un chemin.



Constantes de chaînes de caractères fixées

Les constantes fixes de chaînes de caractères sont disponibles.

Constante chaîne de caractères



Utilisez cette constante pour fournir une chaîne de caractères ASCII constante au diagramme. Définissez cette chaîne de caractères en cliquant dans la constante avec l'outil Doigt et en entrant une valeur. Vous pouvez changer le mode d'affichage pour visualiser les caractères non affichables ou l'équivalent hexa des caractères. Vous pouvez aussi configurer la constante sur le mode d'affichage mot de passe pour afficher des astérisques lorsque vous tapez des caractères.

La valeur de la constante de chaîne de caractères ne peut pas être modifiée pendant l'exécution du VI. Vous pouvez affecter une étiquette à cette constante.

Retour chariot



Chaîne constante contenant la valeur ASCII CR.

Chaîne de caractères vide



Chaîne constante vide. Sa longueur est égale à zéro.

Fin de ligne



Chaîne constante contenant la valeur de fin de ligne, dépendante de la plate-forme. Sous Windows, cette valeur est CRLF ; sous Macintosh, cette valeur est CR ; et sous UNIX, cette valeur est LF.

Retour à la ligne



Chaîne constante contenant la valeur ASCII LF.

Tab

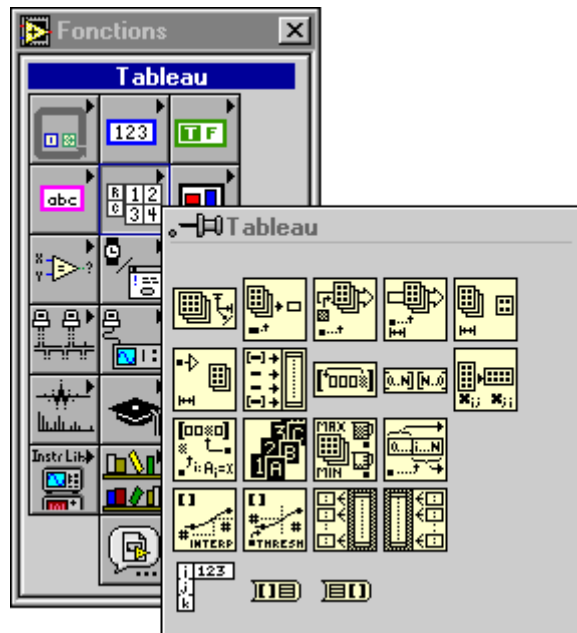


Chaîne constante contenant la valeur ASCII HT (touche de tabulation horizontale).

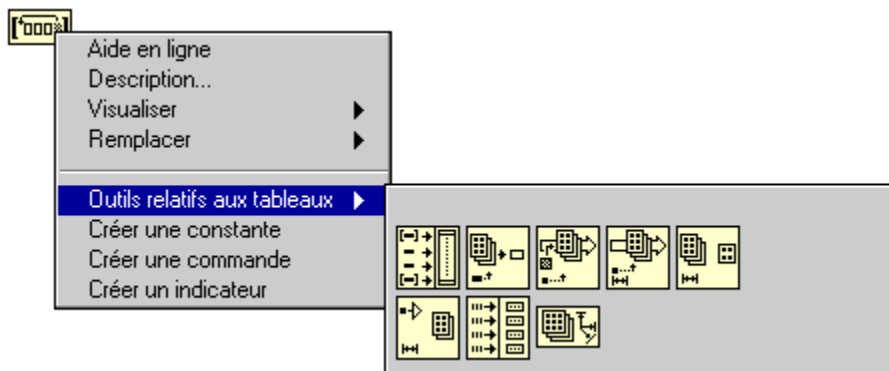
Fonctions de tableaux

Ce chapitre décrit les fonctions utilisées pour les opérations sur les tableaux.

L'illustration suivante montre la palette **Tableau** que vous pouvez ouvrir en sélectionnant **Fonctions»Tableau**.



Quelques fonctions de tableaux sont également disponibles dans la palette **Outils relatifs au tableau** accessible dans la plupart des menus locaux de terminaux. L'illustration ci-dessous montre cette palette.



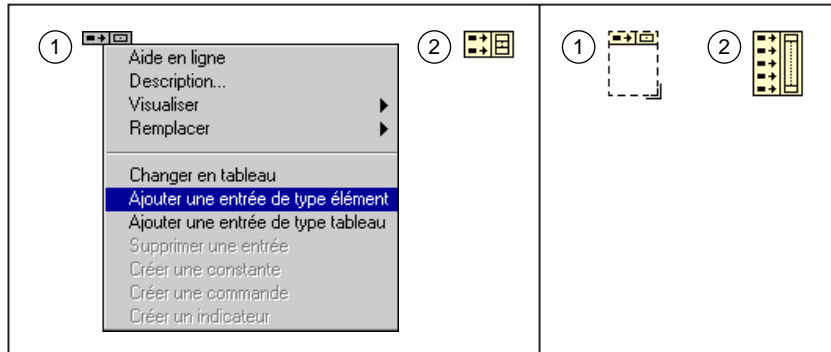
Si vous sélectionnez des fonctions dans cette palette, elles s'affichent avec le nombre correct de terminaux à relier à l'objet sur lequel vous avez ouvert ce menu local.

Pour des exemples sur les fonctions de tableaux, reportez-vous à `exemples\general\array.llb`.

Vue d'ensemble des fonctions de tableaux

Plusieurs des fonctions de tableaux ont un nombre variable de terminaux. Quand vous déposez une nouvelle fonction de ce type, elle apparaît dans le diagramme avec un ou deux terminaux. Vous pouvez ajouter et supprimer des terminaux par le biais des commandes de menu local **Ajouter une entrée de type élément** ou **Ajouter une entrée de type tableau** et **Supprimer une entrée** (dépendant de la fonction) ou en redimensionnant verticalement le nœud depuis un des coins. Si vous souhaitez ajouter des terminaux en ouvrant le menu local, vous devez placer votre pointeur sur les terminaux d'entrée pour ouvrir le menu local.

Vous pouvez réduire le nœud s'il contient des terminaux non câblés. La commande **Ajouter une entrée de type élément** ou **Ajouter une entrée de type tableau** insère un terminal directement après le terminal dont le menu local est ouvert. La commande **Supprimer une entrée** supprime le terminal dont le menu local est ouvert, même s'il est câblé. L'illustration ci-après montre les deux façons d'ajouter des terminaux supplémentaires à la fonction "Construire un tableau".



Valeurs de l'indice de dépassement de gamme

Indexer un tableau au-delà de ses limites génère une valeur par défaut déterminée par le type des éléments du tableau.

Polymorphisme des fonctions de tableaux

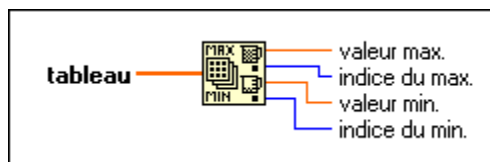
La plupart des fonctions de tableaux acceptent des tableaux de dimensions n de tout type. Cependant, les diagrammes de câblage des descriptions des fonctions montrent des tableaux numériques comme type de données par défaut.

Descriptions des fonctions de tableaux

Les fonctions de tableaux suivantes sont disponibles.

Max & Min d'un tableau

Cherche les premières valeurs maximales et minimales dans un **tableau** numérique. Cette fonction retourne aussi le ou les indice(s) du tableau où résident les valeurs maximales et minimales.

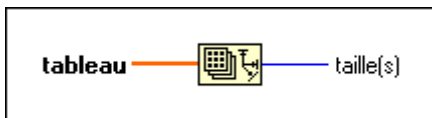


Si le **tableau** numérique est à une dimension, les sorties **indice du max** et **indice du min** sont des entiers scalaires. Si le **tableau** numérique est à plusieurs dimensions, ces sorties sont des tableaux 1D contenant les indices des valeurs maximales et minimales.

La fonction compare chaque type de données selon les règles définies dans le chapitre 9, *Fonctions de comparaison*.

Taille d'un tableau

Retourne le nombre d'éléments contenus dans chaque dimension du **tableau**.



Sous-ensemble d'un tableau

Retourne une portion d'un **tableau** en commençant à **indice** et comportant **longueur** éléments.

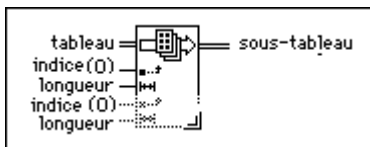
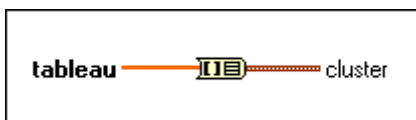


Tableau en cluster

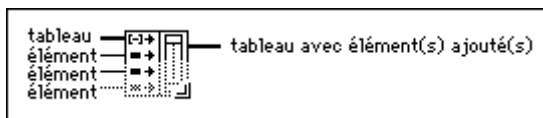
Convertit un tableau 1D en cluster d'éléments du même type que les éléments du tableau. Ouvrez le menu local du nœud pour définir le nombre d'éléments du cluster. La valeur par défaut est neuf. La taille maximale du cluster pour cette fonction est 256.



Pour plus d'informations sur les clusters, reportez-vous au chapitre 8, *Fonctions de clusters*.

Construire un tableau

Assemble n'importe quel nombre de tableaux ou d'éléments d'entrées classés de haut en bas pour créer un **tableau de sortie**.

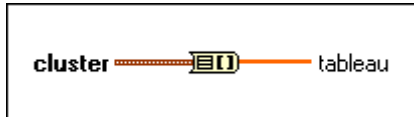


Pour changer une entrée de type d'élément en entrée de type tableau, ouvrez le menu local de l'entrée et sélectionnez **Changer en tableau**. En général, pour construire un tableau de

dimensions n , tous les tableaux doivent avoir la même dimension, n , et chaque élément d'entrée doit avoir pour dimension $n-1$. Pour créer un tableau 1D, connectez des valeurs scalaires aux entrées d'éléments et les tableaux 1D aux entrées de tableau. Pour construire un tableau 2D, connectez des tableaux 1D aux entrées d'élément et les tableaux 2D aux entrées de tableau.

Cluster en tableau

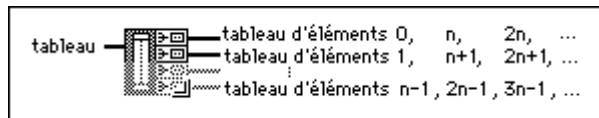
Convertit un cluster de composants de même type en un tableau 1D d'éléments du même type.



Pour plus d'informations sur les clusters, reportez-vous au chapitre 8, *Fonctions de clusters*.

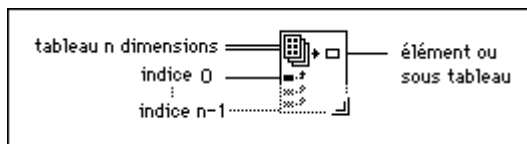
Décimer un tableau 1D

Divise les éléments d'un **tableau** en tableaux de sortie.



Indexer un tableau

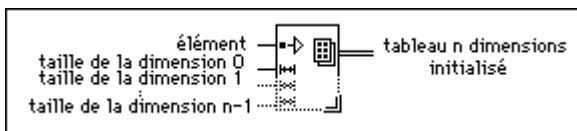
Retourne l'**élément** d'un **tableau** situé à **indice**. Si le **tableau** est multidimensionnel, vous devez ajouter des terminaux **indice** supplémentaires pour chaque dimension de **tableau**.



En plus d'extraire un élément du tableau, vous pouvez découper une composante dimensionnelle d'ordre supérieur en désactivant un ou plusieurs terminaux d'indice.

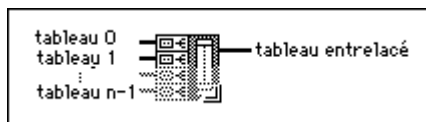
Initialiser un tableau

Construit un tableau de dimensions n dont chaque élément est initialisé à la valeur **élément**.



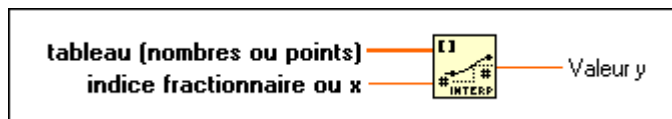
Entrelacer des tableaux 1D

Entrelace des éléments correspondants de tableaux d'entrée en un seul tableau de sortie.



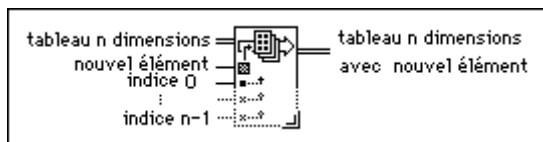
Interpoler un tableau 1D

Utilise la partie entière de l'**indice fractionnel ou x** pour indexer le tableau et la partie fractionnelle de l'**indice fractionnel ou x** pour effectuer une interpolation linéaire entre les valeurs de l'élément indexé et son élément adjacent.



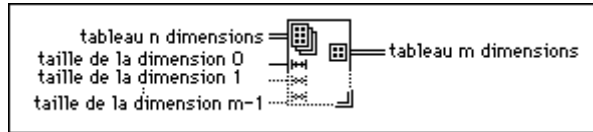
Remplacer un élément d'un tableau

Remplace l'élément d'un **tableau** situé à **indice** par un **nouvel élément**.



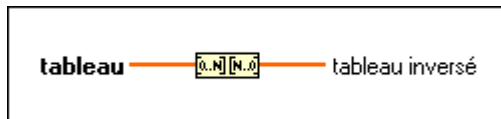
Redimensionner un tableau

Change la dimension d'un tableau selon la valeur de **taille de la dimension**. La fonction est redimensionnable ; **tableau de sortie** comporte une dimension pour chaque entrée de la **taille de la dimension**. Par exemple, vous pouvez utiliser cette fonction pour changer un tableau 1D en tableau 2D ou vice versa. Vous pouvez également l'utiliser pour augmenter ou diminuer la taille d'un tableau 1D.



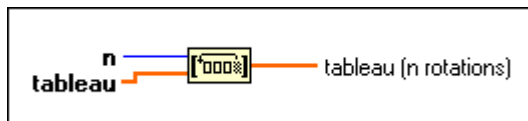
Renverser un tableau 1D

Renverse l'ordre des éléments d'un **tableau**.



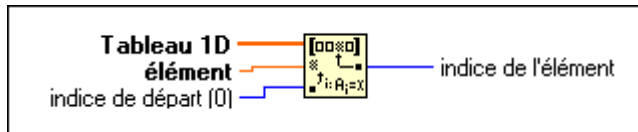
Rotation dans un tableau 1D

Effectue n rotation des éléments d'un **tableau** dans la direction indiquée par le signe de **n**.



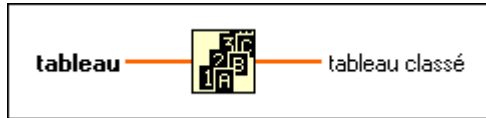
Rechercher dans un tableau 1D

Cherche un **élément** dans un **tableau 1D** à partir d'un **indice de départ**.



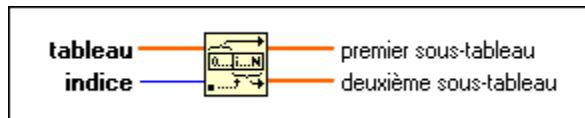
Classer un tableau 1D

Retourne une version d'un **tableau** dont les éléments sont classés dans l'ordre croissant. Les règles de comparaison de chaque type de données sont décrites au chapitre 9, *Fonctions de comparaison*.



Eclater un tableau 1D

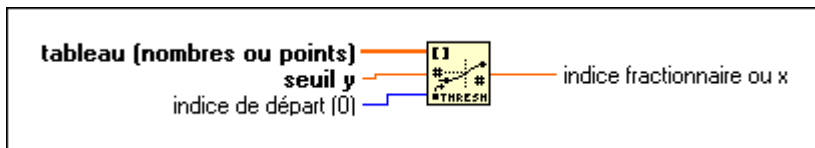
Divise un **tableau** à **indice** et retourne les deux portions.



Seuil d'un tableau 1D

Compare un **seuil y** aux éléments d'un **tableau (nombres ou points)** à partir de **indice de départ** jusqu'à trouver une paire d'éléments consécutifs tels que le **seuil y** soit supérieur à la valeur du premier élément et inférieur ou égal à la valeur du second élément.

La fonction calcule ensuite la distance fractionnelle entre la première valeur et le **seuil y** et retourne l'indice fractionnel correspondant à la position où **seuil y** serait placé dans le **tableau (nombres ou points)** en utilisant une interpolation linéaire.

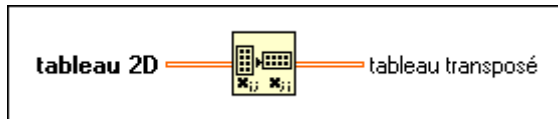


Supposez, par exemple, que le **tableau (nombres ou points)** soit un tableau de quatre chiffres [4, 5, 5, 6], que l'indice **de départ** soit 0 et que le **seuil y** soit 5. L'**indice fractionnel ou x** est alors égal à 1 et correspond à l'indice de la première valeur égale à 5 trouvée par la fonction. Supposez que les éléments d'un tableau soient 6, 5, 5, 7, 6, 6, que l'**indice de départ** soit 0 et que le **seuil y** soit 6 ou moins. La sortie est alors égale à 0. Si le **seuil y** est supérieur à 7 pour le même ensemble de nombres, la sortie est égale à 5. Si le **seuil y** est 14,2, l'**indice de départ** est égale à 5 et les valeurs dans le tableau, à partir d'un indice 5, sont égales à 9,1, 10,3, 12,9 et 15,5, le **seuil y** est situé entre les éléments 7 et 8 car 14,2 est juste entre 12,9 et 15,5. La valeur pour l'**indice fractionnel ou x** est 7,5, c'est-à-dire, à mi-chemin entre 7 et 8.

Si le tableau d'entrée est un tableau de points où chaque point est un cluster de coordonnées x et y , la sortie représente alors la valeur x interpolée correspondant à la position interpolée du **seuil** y au lieu de l'indice fractionnel du tableau. Si la position interpolée du **seuil** y est située juste entre les indices 4 et 5 du tableau avec des valeurs x égales à $-2,5$ et 0 respectivement, la sortie n'est pas une valeur d'indice égale à $4,5$ comme elle serait pour un tableau numérique, mais plutôt une valeur x égale à $-1,25$.

Transposer un tableau 2D

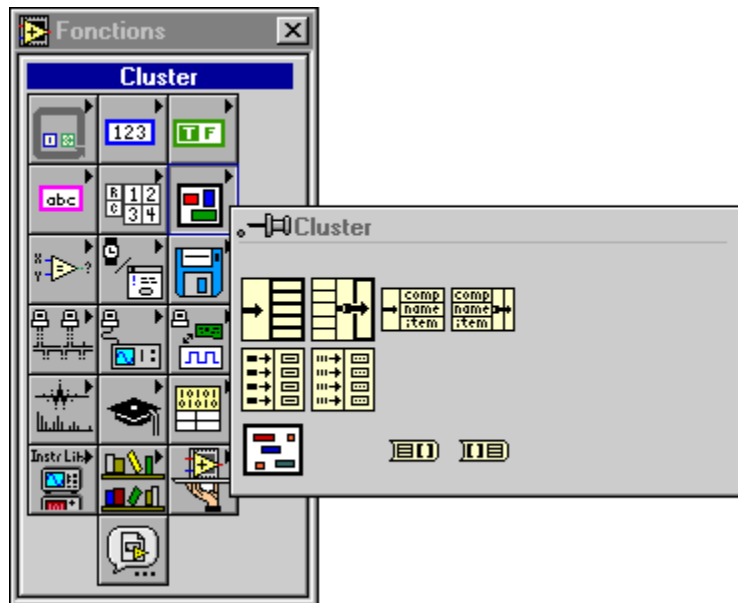
Redispose les éléments d'un **tableau 2D** de façon à ce que le **tableau 2D** $[i,j]$ devienne le **tableau transposé** $[j,i]$.



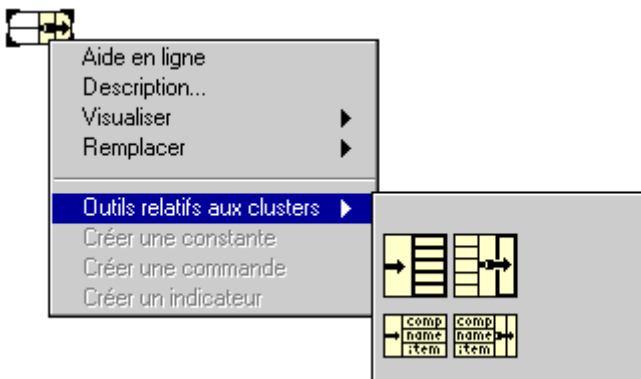
Fonctions de clusters

Ce chapitre décrit les fonctions utilisées pour les opérations sur les clusters.

L'illustration ci-après montre la palette **Cluster** que vous pouvez ouvrir en sélectionnant **Fonctions»Cluster**.



Quelques-unes des fonctions de cluster sont également disponibles dans la palette **Outils relatifs au cluster** de la plupart des menus locaux des terminaux et des fils de liaison. L'illustration suivante montre ce menu local.

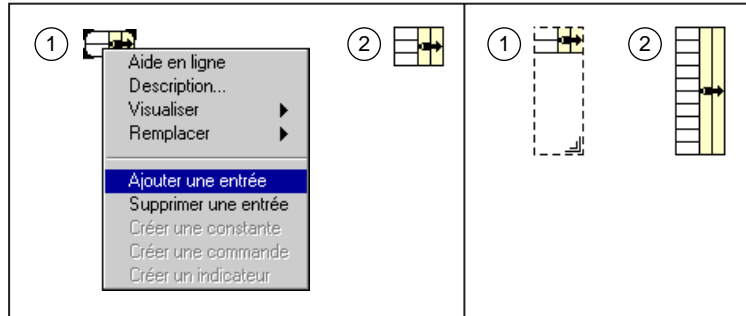


Si vous sélectionnez les fonctions à partir de cette palette, elles apparaissent avec le nombre correct de terminaux à câbler à l'objet sur lequel vous avez ouvert un menu local.

Vue d'ensemble des fonctions de clusters

Quelques fonctions de clusters ont un nombre variable de terminaux. Quand vous déposez une nouvelle fonction de ce type, elle apparaît dans le diagramme avec seulement un ou deux terminaux. Vous pouvez ajouter et supprimer des terminaux avec les options du menu local **Ajouter une entrée** ou **Supprimer une entrée** ou en redimensionnant le nœud avec l'outil Flèche. Si vous souhaitez ajouter des terminaux en ouvrant le menu local, placez votre curseur sur un terminal d'entrée.

Vous pouvez contracter le nœud si l'opération n'efface pas des terminaux déjà câblés. L'option **Ajouter une entrée** insère un terminal directement après le terminal dont vous avez ouvert le menu local. La commande **Supprimer une entrée** supprime le terminal dont vous avez ouvert le menu local, même s'il est câblé. L'illustration suivante montre les deux façons d'ajouter plus de terminaux à la fonction Assembler.



Polymorphisme des fonctions de clusters

Les fonctions Assembler et Désassembler n'affichent pas de type de données sur leurs terminaux d'entrée ou de sortie individuels tant que vous n'avez pas câblé d'objets à ces terminaux. Quand vous les câblez, ces terminaux affichent les types de données des terminaux des commandes ou des indicateurs de la face-avant correspondante.

Etablir l'ordre des éléments d'un cluster

Aux éléments d'un cluster est associé un ordre logique qui n'est pas en rapport avec leurs positions dans le cadre du cluster. Le premier objet que vous insérez du cluster est l'élément 0, le second est l'élément 1, et ainsi de suite. Si vous effacez un élément, l'ordre s'ajuste automatiquement. Vous pouvez changer l'ordre des éléments en sélectionnant l'option **Ordonner le cluster...** dans le menu local du cluster.

Le fait de cliquer sur un élément avec le curseur lui associe le numéro affiché à l'intérieur de la barre d'outils. Vous pouvez modifier ce numéro en tapant un nouveau chiffre dans ce champ. Quand l'ordre souhaité est obtenu, cliquez sur le bouton **Entrée** pour le paramétrer et quittez le mode Edition du cluster. Cliquez sur le bouton **X** pour revenir à l'ancienne configuration.

L'ordre du cluster détermine l'ordre des éléments affichés sur les terminaux du diagramme des fonctions Assembler et Désassembler.

Les fonctions "Assembler par nom" et "Désassembler par nom" permettent un accès plus facile aux données d'un cluster. Avec ces fonctions, vous pouvez accéder par nom à des éléments spécifiques d'un cluster et n'accéder ainsi qu'aux éléments souhaités. Vu que ces fonctions se rapportent au nom des composantes et non à leur position, vous pouvez

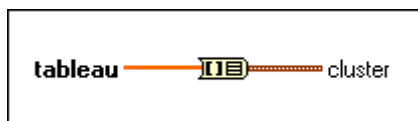
changer la structure de données d'un cluster sans briser de liens tant que vous ne changez pas le nom ni ne supprimez de composantes du cluster.

Descriptions des fonctions de clusters

Les fonctions de clusters suivantes sont disponibles.

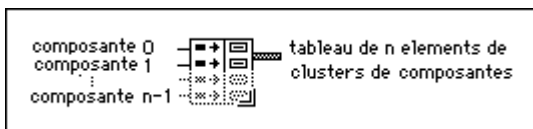
Tableau en cluster

Convertit un tableau 1D en cluster d'éléments du même type que les éléments de tableau. Ouvrez le menu local du nœud pour paramétrer le nombre d'éléments du cluster. La valeur par défaut est neuf. La taille maximale du cluster pour cette fonction est 256.



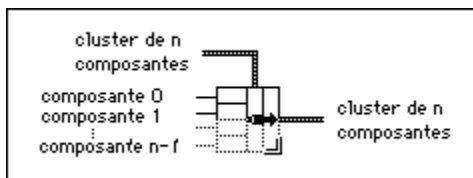
Construire un tableau de clusters

Assemble toutes les entrées **composantes** de haut en bas en un tableau de clusters de **composante**. Si les entrées correspondent à quatre composantes simple précision à virgule flottante, la sortie est un tableau de clusters de quatre éléments contenant un nombre simple précision à virgule flottante. L'élément 0 du tableau a la valeur de la composante du haut, et ainsi de suite.



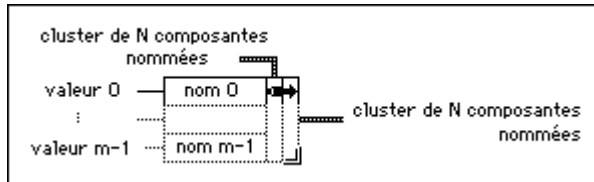
Assembler

Assemble toutes les composantes d'entrée en un seul cluster.



Assembler par nom

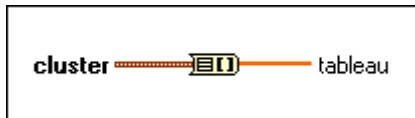
Remplace les composantes d'un cluster existant. Une fois que vous avez câblé le terminal d'entrée de cluster, vous pouvez ouvrir le menu local des terminaux de noms pour faire un choix dans la liste des composantes du cluster.



Vous devez toujours câbler l'entrée **cluster**. Si vous créez un cluster pour un indicateur de cluster, vous pouvez relier une variable locale de cet indicateur à l'entrée **cluster**. Si vous créez un cluster pour une commande de cluster d'un sous-VI, vous pouvez placer une copie de cette commande (peut-être masquée) sur la face-avant du VI et câbler la commande à l'entrée du **cluster**.

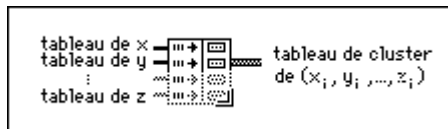
Cluster en tableau

Convertit un cluster de composantes du même type en un tableau 1D d'éléments du même type.

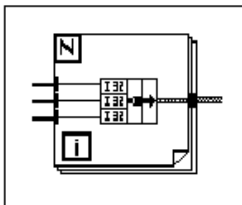


Indexer & Assembler un tableau de clusters

Indexe un ensemble de tableaux et crée un tableau de clusters dont le i ème élément contient le i ème élément de chaque tableau d'entrée.

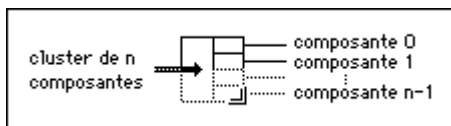


Cette fonction équivaut au diagramme suivant et convertit un cluster de tableaux en un tableau de clusters.



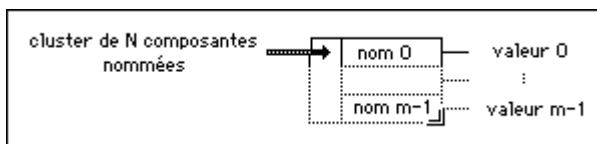
Désassembler

Désassemble un cluster en composantes individuelles.



Désassembler par nom

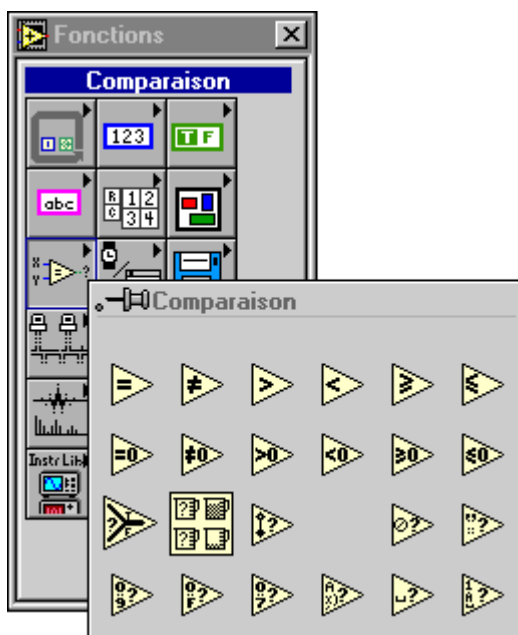
Retourne les éléments d'un cluster dont vous spécifiez les noms. Vous pouvez sélectionner l'élément auquel vous souhaitez accéder en ouvrant le menu local des terminaux de sortie et en sélectionnant un nom dans la liste des éléments du cluster.



Fonctions de comparaison

Ce chapitre décrit les fonctions utilisées pour effectuer des comparaisons ou des tests conditionnels.

L'illustration suivante montre la palette **Comparaison** que vous pouvez ouvrir en sélectionnant **Fonctions»Comparaison**.



Vue d'ensemble des fonctions de comparaison

Cette section présente les fonctions de comparaison.

Comparaison booléenne

Les fonctions de comparaison interprètent la valeur booléenne Vrai (TRUE) comme étant supérieure à la valeur booléenne Faux (FALSE).

Comparaison de chaînes de caractères

Ces fonctions comparent des chaînes de caractères à partir de l'équivalent numérique de leur code ASCII. C'est ainsi que le caractère a (dont la valeur décimale est 97) est supérieur à A (65), qui est lui-même supérieur au chiffre 0 (48), qui est supérieur au caractère d'espace (32). Ces fonctions comparent les caractères un par un depuis le début de la chaîne jusqu'à ce qu'une inégalité surgisse, provoquant la fin de la comparaison. Par exemple, LabVIEW compare les chaînes abcd et abef jusqu'à c, dont la valeur est inférieure à celle de e. La présence d'un caractère est supérieure à l'absence d'un caractère. Ainsi, la chaîne de caractères abcd est supérieure à la chaîne de caractères abc car la première est plus longue.

Les fonctions qui testent la catégorie du caractère d'une chaîne (par exemple, les fonctions Chiffre décimal ? et Imprimable ?) n'évaluent que le premier caractère de la chaîne.

Comparaison numérique

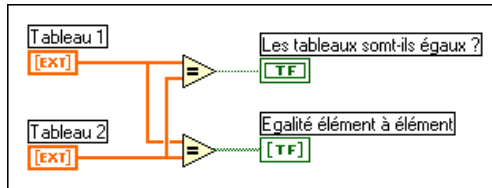
La plupart des fonctions de comparaison testent une entrée ou comparent deux entrées et retournent une valeur booléenne. Les fonctions convertissent les nombres à la même représentation avant de les comparer. Les comparaisons avec des valeurs qui ne sont pas des nombres (NaN) retournent une valeur qui indique une inégalité.

Comparaison de clusters

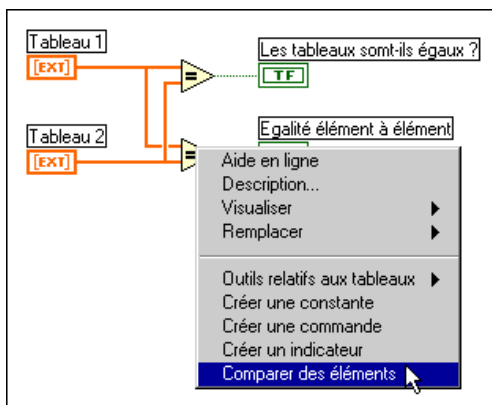
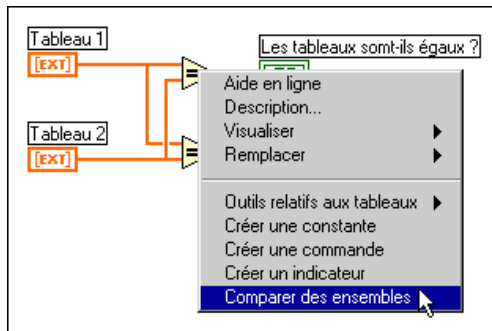
Des fonctions de comparaison comparent des clusters de la même façon que les chaînes de caractères, un élément à la fois en commençant par l'élément 0 jusqu'à ce qu'une inégalité surgisse. Les clusters doivent avoir le même nombre d'éléments, des éléments du même type et classés dans le même ordre si vous souhaitez les comparer.

Modes de comparaison

Quelques fonctions ont deux modes de comparaison de tableaux ou de clusters. Si vous comparez deux tableaux ou clusters en mode **Comparer des ensembles**, la fonction retourne une valeur simple. En mode **Comparer des éléments**, la fonction compare les éléments individuellement et retourne ensuite un tableau ou un cluster de valeurs booléennes. L'illustration suivante montre les deux modes.



Vous pouvez changer le mode de comparaison en sélectionnant **Comparer des éléments** ou **Comparer des ensembles** dans le menu local du nœud, comme l'indiquent les illustrations suivantes.



Lorsque vous comparez deux tableaux de longueurs différentes en mode **Comparer des éléments**, LabVIEW ignore les éléments du grand tableau dont l'indice est supérieur à celui du dernier élément du petit tableau.

La procédure pour comparer deux tableaux en mode **Comparer des ensembles** est la suivante : (1) LabVIEW cherche le premier ensemble d'éléments correspondant dans les deux entrées qui diffèrent et les utilise pour déterminer les résultats de la comparaison ; (2) si tous les éléments sont identiques mais qu'un tableau comporte plus d'éléments, LabVIEW considère que le tableau plus long est supérieur au tableau plus court ; (3) si tous les éléments sont les mêmes dans les deux tableaux et que les tableaux sont de la même longueur, les tableaux sont identiques. Ainsi, LabVIEW considère que le tableau [1,2,3] est supérieur au tableau [1,2] et retourne une valeur booléenne simple en mode **Comparer des ensembles**.

Les tableaux doivent avoir le même nombre de dimensions (par exemple, les deux doivent être bidimensionnels) et, pour une comparaison consécutive entre des tableaux à plusieurs dimensions, chaque dimension doit être de la même taille.

Pour les clusters en mode **Comparer des ensembles**, LabVIEW effectue la comparaison selon l'ordre du cluster. Les deux clusters que LabVIEW compare doivent avoir le même nombre d'éléments.

Certaines fonctions de comparaison ne disposent pas des modes **Comparer des ensembles** ou **Comparer des éléments**. Elles comparent les tableaux de la même façon que les chaînes de caractères : un élément à la fois en commençant par l'élément 0 jusqu'à ce qu'une inégalité surgisse.

Comparaison de caractères

Vous pouvez utiliser les fonctions de comparaison de caractères pour déterminer le type de caractère. Les fonctions suivantes sont des fonctions de comparaison de caractères.

- Chiffre décimal ?
- Chiffre hexa ?
- Classe lexicale
- Chiffre octal ?
- Imprimable ?
- Espace blanc ?

Si leur entrée est une chaîne de caractères, les fonctions testent le premier caractère. Si leur entrée est une chaîne vide, le résultat est Faux (FALSE). Si leur entrée est un nombre, les fonctions l'interprètent comme le code d'un caractère ASCII.

Reportez-vous à l'Annexe C, *Messages d'interface multiligne GPIB*, pour obtenir la liste des codes correspondant à chaque caractère ASCII.

Polymorphisme des fonctions de comparaison

Les fonctions “Egax ?”, “Non égaux ?” et Sélectionner acceptent des entrées de tout type mais toutes les entrées doivent être du même type.

Les fonctions “Supérieur ou égal ?”, “Inférieur ou Egal ?”, “Inférieur ?”, “Supérieur ?”, “Max & Min” et “Dans la gamme ?” acceptent des entrées de tout type, à l'exception du type complexe, chemin ou refnum. Toutes les entrées doivent être du même type. Vous pouvez comparer des nombres, des chaînes de caractères, des booléens, des tableaux de chaînes de caractère, des clusters de nombres, des clusters de chaînes de caractères, et ainsi de suite. Vous ne pouvez pas comparer un nombre à une chaîne de caractères ou une chaîne de caractères à un booléen, etc.

Les fonctions qui comparent des valeurs à zéro acceptent des scalaires numériques, des clusters et des tableaux de nombres. Ces fonctions produisent des valeurs booléennes comme sorties dans la même structure de données que leur entrée.

La fonction “Pas un nombre/chemin/refnum” accepte les mêmes types d'entrées que les fonctions qui comparent des valeurs à zéro. Cette fonction accepte aussi des chemins et des refnums. “Pas un nombre/chemin/refnum” produit des valeurs booléennes dans la même structure de données que son entrée. Reportez-vous au chapitre 11, *Fonctions des fichiers* et au chapitre 31, *Introduction aux VIs d'E/S d'instruments LabVIEW*, pour plus d'informations sur ces fonctions.

Les fonctions “Chiffre décimal ?”, “Chiffre hexa ?”, “Chiffre octal ?”, “Imprimable ?” et “Espace blanc ?” acceptent une chaîne de caractères ou des nombres scalaires, des clusters de chaînes de caractères ou de nombres non complexes, des tableaux de chaînes de caractères ou de nombres non complexes, et ainsi de suite. La sortie produit des valeurs booléennes dans la même structure de données que l'entrée.

La fonction “Chemin/chaîne de caractères vide ?” accepte des chemins, des chaînes de caractères scalaires, des clusters de chaînes, des tableaux de chaînes, et ainsi de suite. La sortie produit des valeurs booléennes dans la même structure de données que l’entrée.

Vous pouvez utiliser les fonctions Egaux ?, Non Egaux ?, “Pas un nombre/chemin/refnum ?”, “Chemin/chaîne de caractères vide ?” et Sélectionner avec des chemins et des refnums. Cependant, aucune autre fonction de comparaison n’accepte des chemins ou des refnums comme entrées.

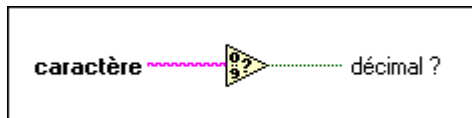
Les fonctions de comparaison qui utilisent des tableaux et des clusters produisent normalement des tableaux booléens et des clusters de la même structure. Vous pouvez ouvrir le menu local et sélectionner **Comparer des ensembles**. Dans ce cas, la fonction produit une valeur booléenne simple comme sortie. La fonction compare des ensembles en comparant le premier ensemble d’éléments pour produire une sortie. Si les premiers éléments sont égaux, la fonction compare le second ensemble d’éléments, et ainsi de suite.

Descriptions des fonctions de comparaison

Les fonctions de comparaison suivantes sont disponibles.

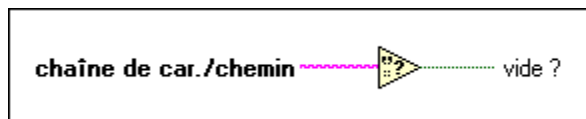
Chiffre Décimal ?

Retourne Vrai (TRUE) si **caractère** est un chiffre décimal compris entre 0 et 9. Sinon, cette fonction retourne Faux (FALSE).



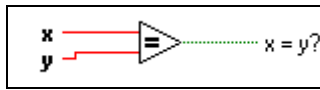
Chemin/Chaîne de caractères vide ?

Retourne Vrai (TRUE) si **chemin/chaîne de car.** est un chemin ou une chaîne vide. Sinon, cette fonction retourne Faux (FALSE).



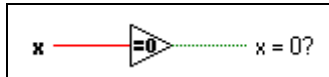
Egax ?

Retourne Vrai (TRUE) si x est égal à y . Sinon, cette fonction retourne Faux (FALSE).



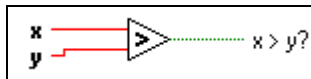
Egal à 0 ?

Retourne Vrai (TRUE) si x est égal à 0. Sinon, cette fonction retourne Faux (FALSE).



Supérieur ?

Retourne Vrai (TRUE) si x est supérieur à y . Sinon, cette fonction retourne Faux (FALSE).



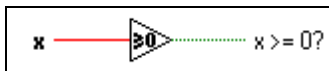
Supérieur ou égal ?

Retourne Vrai (TRUE) si x est supérieur à ou égal à y . Sinon, cette fonction retourne Faux (FALSE).



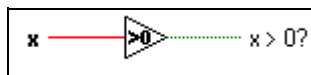
Supérieur ou égal à 0 ?

Retourne Vrai (TRUE) si x est supérieur à ou égal à 0. Sinon, cette fonction retourne Faux (FALSE).



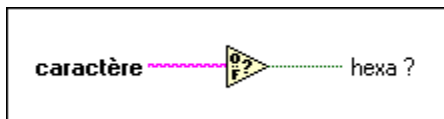
Supérieur à 0 ?

Retourne Vrai (TRUE) si x est supérieur à 0. Sinon, cette fonction retourne Faux (FALSE).



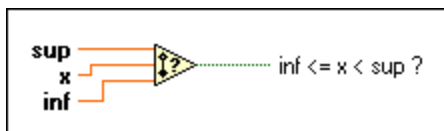
Chiffre hexa ?

Retourne Vrai (TRUE) si **caractère** est un chiffre hexa compris entre 0 et 9, A et F ou a et f. Sinon, cette fonction retourne Faux (FALSE).



Dans la gamme ?

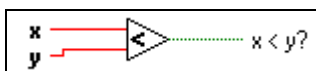
Retourne Vrai (TRUE) si **x** est supérieur ou égal à **inf** et inférieur à **sup**. Sinon, cette fonction retourne Faux (FALSE).



Remarque *Cette fonction s'exécute toujours en mode Comparer les ensembles. Pour créer un tableau booléen comme sortie, vous devez exécuter cette fonction dans une structure en boucle.*

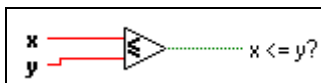
Inférieur ?

Retourne Vrai (TRUE) si **x** est inférieur à **y**. Sinon, cette fonction retourne Faux (FALSE).



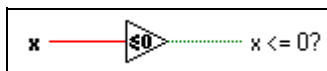
Inférieur ou égal ?

Retourne Vrai (TRUE) si **x** est inférieur ou égal à **y**. Sinon, cette fonction retourne Faux (FALSE).



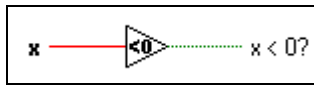
Inférieur ou égal à 0 ?

Retourne Vrai (TRUE) si **x** est inférieur ou égal à 0. Sinon, cette fonction retourne Faux (FALSE).



Inférieur à 0 ?

Retourne Vrai (TRUE) si x est inférieur à 0. Sinon, cette fonction retourne Faux (FALSE).



Classe lexicale

Retourne le **numéro de classe** du **caractère**.

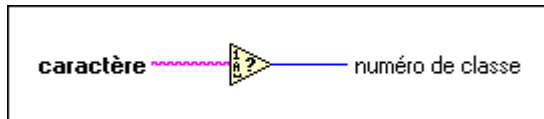
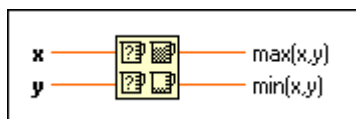


Tableau 9-1. Descriptions des numéros de classe lexicale

Numéro de classe	Classe lexicale
0	Caractères étendus avec un préfixe de touche de commande ou d'option (codes 128 à 255)
1	Caractères ASCII non affichables (codes 0 à 31, excepté 9 à 13)
2	Caractères en blanc : Espace, Tab, Retour du chariot, Saut de page, Nouvelle ligne et Tab vertical (codes 32, 9, 13, 12, 10 et 11, respectivement)
3	Chiffres 0 à 9
4	Caractères en majuscules A à Z
5	Caractères en minuscules a à z
6	Tous les caractères non alphanumériques ASCII imprimables

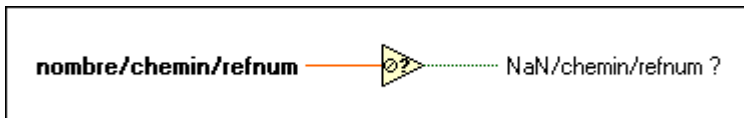
Max & Min

Compare x et y et retourne la plus grande valeur sur le terminal de sortie supérieur et la plus petite valeur sur le terminal de sortie inférieur.



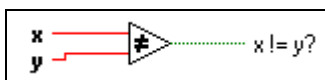
Pas un nombre/chemin/refnum ?

Retourne Vrai (TRUE) si **nombre/chemin/refnum** est égal à Pas un nombre (NaN), Pas un Chemin ou Pas un Refnum. Sinon, cette fonction retourne Faux (FALSE). NaN peut être le résultat d'une division par 0, la racine carrée d'un nombre négatif, etc.



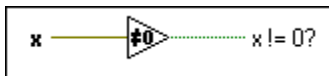
Non égaux ?

Retourne Vrai (TRUE) si **x** n'est pas égal à **y**. Sinon, cette fonction retourne Faux (FALSE).



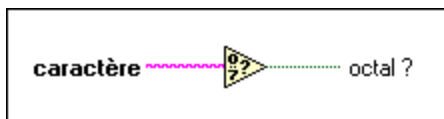
Non égal à 0 ?

Retourne Vrai (TRUE) si **x** n'est pas égal à 0. Sinon, cette fonction retourne Faux (FALSE).



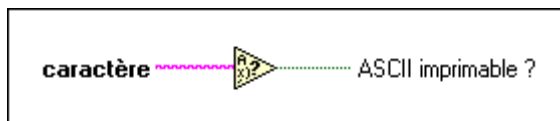
Chiffre octal ?

Retourne Vrai (TRUE) si **caractère** est un chiffre octal compris entre 0 et 7. Sinon, cette fonction retourne Faux (FALSE).



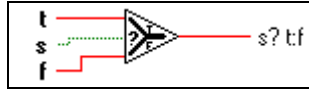
Imprimable ?

Retourne Vrai (TRUE) si **caractère** est un caractère ASCII imprimable. Sinon, cette fonction retourne Faux (FALSE).



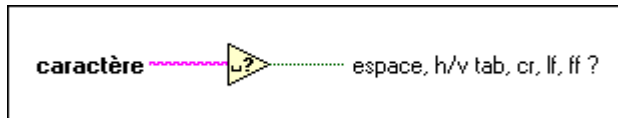
Sélectionner

Retourne la valeur connectée à l'entrée **t** ou **f**, selon la valeur de **s**. Si **s** est Vrai (TRUE), cette fonction retourne la valeur connectée à **t**. Si **s** est Faux (FALSE), cette fonction retourne la valeur connectée à **f**.



Espace blanc ?

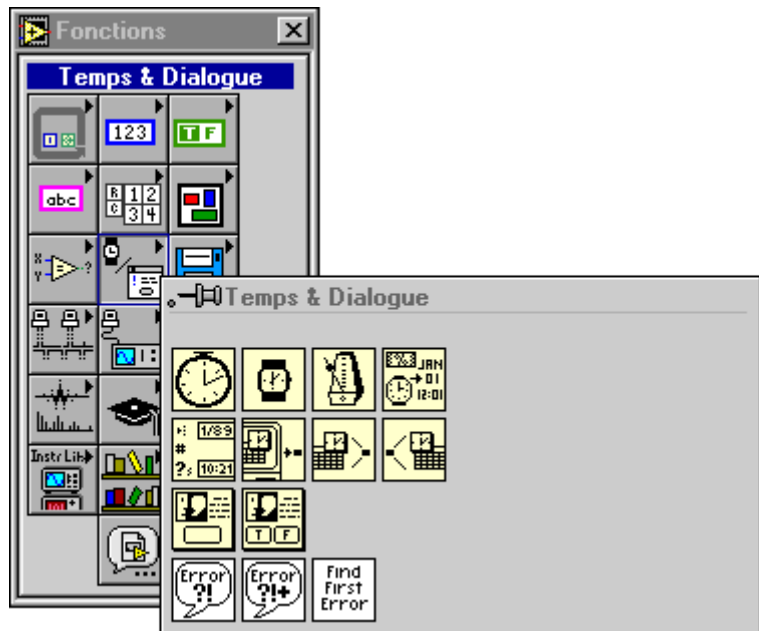
Retourne Vrai (TRUE) si **caractère** est un caractère en blanc, comme Espace, Tab, Nouvelle ligne, Retour chariot, Saut de page ou Tab vertical. Sinon, la fonction retourne Faux (FALSE).



Fonctions de temps, de dialogue et d'erreurs

Ce chapitre décrit les fonctions temporelles, utilisées pour connaître l'heure courante, mesurer le temps écoulé ou suspendre une opération pour une durée spécifique. La gestion d'erreurs est également discutée dans ce chapitre.

L'illustration suivante montre la palette **Temps & dialogue** que vous pouvez ouvrir en sélectionnant **Fonctions»Temps & dialogue**.



Vue d'ensemble des fonctions de temps, de dialogue et d'erreurs

Cette section présente les fonctions de temps, de dialogue et d'erreurs.

Fonctions temporelles

Les fonctions “Date/Heure en secondes” et “Secondes en Date/Heure” ont un paramètre nommé **cluster date/heure**, constitué d'entiers de 32 bits classés dans l'ordre suivant.

Tableau 10-1. Valeur valide des éléments du cluster Date/heure

	Élément	Valeurs valables
0	(seconde)	0 à 59
1	(minute)	0 à 59
2	(heure)	0 à 23
3	(jour du mois)	1 à 31 comme sortie de la fonction ; 1 à 366 comme entrée
4	(mois)	1 à 12
5	(année)	1904 à 2040
6	(jour de la semaine)	1 à 7 (dimanche à samedi)
7	(jour de l'année)	1 à 366
8	(DST)	0 à 1 (0 pour Heure standard, 1 Heure d'été)

Les fonctions “Attendre (ms)” et “Attendre un multiple de ms” exécutent des appels asynchrones au système, bien que les nœuds fonctionnent de manière synchrone. Ainsi, elles n'achèvent leur exécution qu'après une durée spécifiée. Les fonctions utilisent les appels asynchrones pour que les autres nœuds puissent s'exécuter pendant que les nœuds temporels attendent.



Remarque *Les valeurs temporelles hors de la gamme comprise entre 2082844800 et 4230328447 secondes ou 00:00, 1 jan., 1970, Heure universelle à 3:14, 19 jan., 2038, Heure universelle peuvent ne pas être converties dans les mêmes dates sur toutes les plates-formes. Cette exception existe principalement sous Windows 3.x, qui ne supporte pas les dates avant le 1 jan., 1970, Heure universelle.*

Vue d'ensemble de la gestion d'erreurs

Chaque fois que vous élaborez un programme, envisagez la possibilité d'une erreur et la façon dont le programme traitera ce problème. LabVIEW ne vous avertit automatiquement par une boîte de dialogue que lorsque quelques erreurs d'exécution surviennent, le plus souvent dans le cas des opérations sur les boîtes de dialogue de fichiers. LabVIEW n'indique cependant pas toutes les erreurs. Si le système indiquait toutes les erreurs, vous n'auriez plus la flexibilité nécessaire pour savoir que faire lorsqu'une erreur se produit et comment en avertir l'utilisateur.

Une vérification rigoureuse des erreurs, tout particulièrement pour les opérations d'E/S (fichier, série, GPIB, acquisition de données et communication) est essentielle à toutes les phases d'un projet. Cette section décrit trois situations d'E/S où des erreurs peuvent survenir.

Un premier type d'erreur peut survenir lorsque vous avez mal initialisé vos communications ou transmis des données incorrectes à votre périphérique externe. Ce type de problème survient en général pendant la conception du programme et disparaît quand vous terminez la mise au point. Cependant, si vous n'incorporez pas de moyen de vérification d'erreurs, le repérage d'une simple erreur de programmation peut exiger énormément de temps. Sans vérification d'erreurs, vous savez uniquement que votre programme ne s'exécute pas. Vous ne savez pas pourquoi l'erreur est survenue et où en est la source.

Un second type d'erreur peut survenir lorsque votre périphérique externe est éteint, en panne ou incapable d'exécuter ses fonctions habituelles. Ce type de problème peut surgir à tout moment, mais si une vérification d'erreurs est incorporée, votre programme vous avertit dès qu'un tel échec opérationnel survient.

Le troisième type d'erreur peut survenir quand vous effectuez la mise à jour de LabVIEW ou de votre système d'exploitation et que vous provoquez un bogue du programme G ou du programme du système. Ce type d'erreur signifie qu'il faut vérifier des erreurs jusque-là ignorées (par exemple, les fonctions qui ferment des fichiers ou qui réinitialisent l'acquisition de données). Vérifiez la présence d'erreurs dans toutes les opérations d'E/S.

Il peut paraître plus facile d'ignorer la vérification d'erreur quand vous devez ajouter un code de gestion d'erreur pour tester et indiquer les erreurs. Les VIs décrits ici sont conçus pour faciliter la création de programmes qui incorporent la vérification et la gestion d'erreurs.

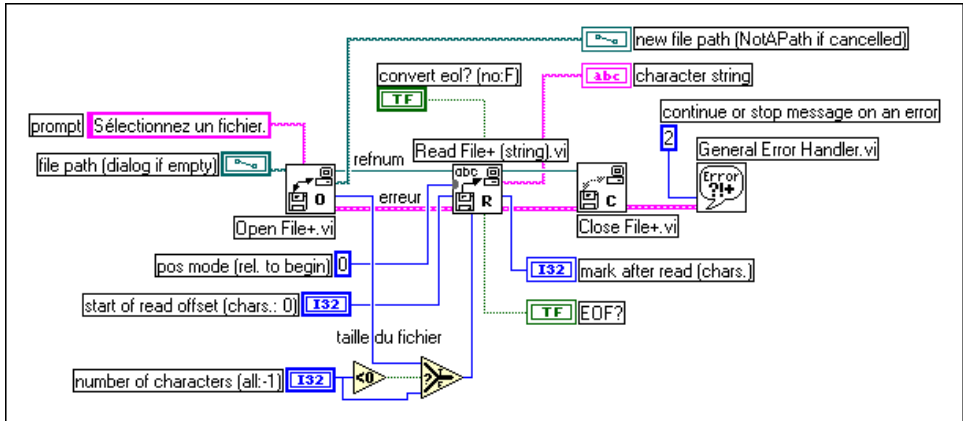
Les fonctions G et des VIs de bibliothèque retournent des erreurs de deux façons : avec des codes d'erreur numériques ou avec un cluster d'état d'erreur. En général, les fonctions produisent des codes d'erreur de sortie tandis que les VIs incorporent le cluster d'erreurs nommés entrée/sortie d'erreurs (E/S d'erreurs).

E/S d'erreur et cluster d'état d'erreur

Le concept d'E/S d'erreur est logique pour l'architecture du flux de données en G. Si les informations des données peuvent être transmises entre des nœuds, les informations des états d'erreur le peuvent également. Chaque nœud, nécessitant des informations d'erreur, teste l'état d'erreur reçue et réagit en conséquence. Si aucune erreur ne s'est produite, le nœud s'exécute normalement. Si une erreur s'est produite, le nœud la détecte, saute l'exécution et transmet l'état d'erreur au nœud suivant, qui réagit de la même manière. De cette façon, la notification sur la première erreur qui survient lors d'une séquence d'opérations est transmise à tous les nœuds et chaque nœud réagit face à l'erreur. A la fin du flux de données, votre programme avertit l'utilisateur qu'une erreur est survenue.

L'E/S d'erreur présente un autre avantage : vous pouvez vous en servir pour contrôler l'ordre d'exécution d'opérations indépendantes. Alors que vous pouvez utiliser un taskID DAQ pour contrôler l'ordre des opérations DAQ pour un groupe, vous ne pouvez pas l'utiliser pour contrôler l'ordre de plusieurs groupes. Un taskID DAQ ne fonctionne pas avec d'autres types d'opérations d'E/S telles que les opérations de fichiers.

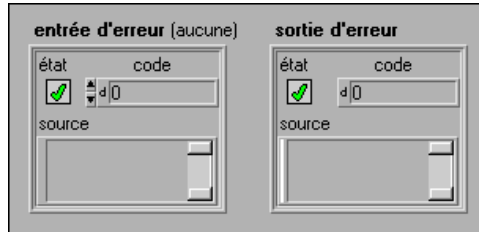
Le diagramme suivant du VI Utilitaire, "Lire les caractères à partir du fichier" (`Read characters from File.vi`), montre comment une E/S d'erreur est implémentée dans un simple VI.



L'opération commence au niveau du VI "Ouvrir un fichier +" (Open file +.vi). Si le fichier est correctement ouvert, "Lire un fichier +(chaîne de caractères)" (Read file +(string).vi) lit le fichier et "Fermer un fichier +" (Close File +.vi) le ferme. Si vous définissez un chemin non valable, "Ouvrir un fichier +" (Open file+.vi) détecte l'erreur et transmet l'état d'erreur à travers les deux autres VIs au "Gestionnaire général d'erreurs", qui émet un avertissement. Observez que la gestion d'erreurs dans ce diagramme est constituée par le fil de liaison d'erreur et le "Gestionnaire général d'erreurs". Elle n'est ni encombrante ni gênante.

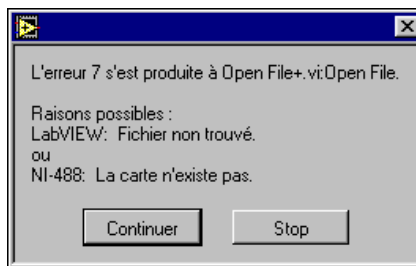
L'état d'erreur comporte trois ensembles d'informations combinés en un cluster d'erreur. L'**Etat** est une valeur booléenne : Vrai (TRUE) si une erreur s'est produite et Faux (FALSE) sinon. Le **Code** est un entier 32 bits signé qui identifie l'erreur. Un **Code** d'erreur différent de zéro avec un **Etat** d'erreur Faux (FALSE) constitue une mise en garde non considérée comme une erreur. Par exemple, un événement de timeout DAQ (code 10800) est normalement indiqué comme mise en garde. La **Source** est une chaîne de caractères qui identifie où l'erreur est survenue.

Les clusters d'états **entrée d'erreur** et **sortie d'erreur** du VI "Ouvrir un fichier+" (Open file+.vi) sont présentés dans l'illustration suivante. Le cluster **entrée d'erreur**, dont la valeur par défaut est *pas d'erreur* n'exige pas de câblage s'il constitue le premier de la chaîne.



Vous pouvez trouver les clusters **entrée d'erreur** et **sortie d'erreur** en sélectionnant **Commandes** » **Tableau & Cluster** sur la face-avant.

L'illustration suivante montre le message transmis par le "Gestionnaire général d'erreurs" si vous définissez un chemin non valable.



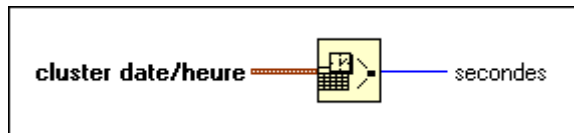
Le "Gestionnaire général d'erreurs" constitue un des trois VIs utilitaires utilisés pour traiter les erreurs. Il comprend une base de données des codes et des descriptions d'erreurs pour créer des messages comme celui montré plus haut. Le "Gestionnaire d'erreurs simple" exécute la même opération de base mais dispose de moins d'options. Le troisième VI, "Trouver la première erreur (Find First Error.vi)", crée un cluster d'E/S d'erreur à partir de fonctions ou de VIs qui ne produisent que des codes d'erreurs scalaires.

Descriptions des fonctions de temps et de dialogue

Les fonctions suivantes de temps et de dialogue sont disponibles.

Date/heure en secondes

Convertit un cluster de neuf entiers 32 bits signés spécifiant l'heure locale (seconde, minute, heure, jour, mois, année, jour de la semaine, jour de l'année et heure standard ou d'été) dans la zone horaire configurée pour votre ordinateur en un nombre de **secondes** indépendant de la zone horaire écoulées depuis le vendredi 1^{er} janvier, 1904 à 00:00, heure universelle.

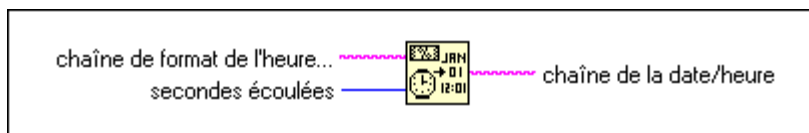


Les entiers du jour de la semaine, jour de l'année et heure d'été sont ignorés. Si n'importe quel autre entier est hors des gammes spécifiées dans le tableau 10-1, les résultats sont imprévisibles.

Quand il est utilisé comme entier, le jour du mois doit être compris dans la gamme 1 à 366. Ainsi, vous pouvez spécifier des dates du calendrier Julien en paramétrant le mois sur janvier et le jour actuel sur le jour de l'année. Par exemple, utilisez janvier 150 pour le 150^{ème} jour de l'année.

Format de la chaîne de caractères de la date/heure

Vous permet d'afficher la date et l'heure sous un format que vous spécifiez.



La **chaîne de la date/heure** est calculée en fonction de l'entrée **secondes (maintenant)**, qui représente le nombre de secondes écoulées depuis le 1^{er} janvier 1904 à 00:00, heure universelle, et la **chaîne de format de l'heure** définit le format de la chaîne de sortie.

Si **secondes** n'est pas câblé, l'heure actuelle est utilisée. Si **chaîne de format de l'heure** n'est pas câblée, l'option par défaut est %c, ce qui correspond à la représentation locale de la date/heure.

“Formater la chaîne des caractères de la date/heure” calcule la **chaîne de la date/heure** en copiant la **chaîne de format de l'heure** et en remplaçant chaque code de format par la valeur correspondante du tableau suivant.

Tableau 10-2. Codes de format pour la chaîne de format de l'heure

Code de format	Valeur
%%	un simple caractère de pourcentage
%a	nom abrégé du jour de la semaine (par ex., mer)
%A	nom complet du jour de la semaine (par ex., mercredi)
%b	nom abrégé du mois (par ex., jan)
%B	nom complet du mois (par ex., janvier)
%c	représentation locale par défaut de la date et de l'heure
%d	jour du mois (01–31)
%H	heure (horloge de 24 heures) (00–23)
%I	heure (horloge de 12 heures) (01–12)
%j	numéro du jour de l'année (001–366)
%m	numéro du mois (01–12)
%M	minute (00–59)
%P	indicateur AM ou PM
%S	secondes (00–59)
%U	numéro de la semaine de l'année (00–53), où dimanche correspond au premier jour de la semaine
%w	jour de la semaine comme chiffre décimal (0–6), où 0 représente dimanche
%W	numéro de la semaine de l'année (00–53), où lundi correspond au premier jour de la semaine
%x	représentation locale de la date
%X	représentation locale de la date
%y	année d'un siècle (00–99)
%Y	année, siècle compris (par ex., 1997)
%Z	nom ou abréviation de la zone horaire

Les caractères qui apparaissent dans une **chaîne de format de l'heure** et qui ne font pas partie d'un code de format sont copiés tels quels sur la chaîne de caractères de sortie. Les codes de format de l'heure (commençant par %) qui ne sont pas reconnus sont également copiés dans la chaîne de sortie.

Les codes de format de l'heure sont précédés de zéro si nécessaire pour assurer une largeur constante du champ. Un modificateur # optionnel placé avant la lettre du code de format supprime les zéros précédant les codes de format suivants :

`%#d, %#H, %#I, %#j, %#m, %#M, %#S, %#S, %#U, %#w, %#W, %X, %#y, %#Y.`

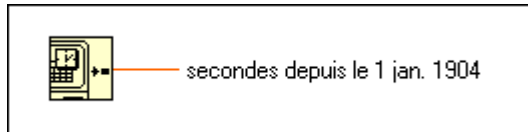
Le modificateur # n'affecte pas le comportement de tout autre code de format.



Remarque *Les codes de format %c, %x, %X et %Z varient selon le support du système d'exploitation local ; la sortie de ces codes dépend de la plate-forme. L'interprétation de la règle sur l'heure d'été varie aussi entre les plates-formes.*

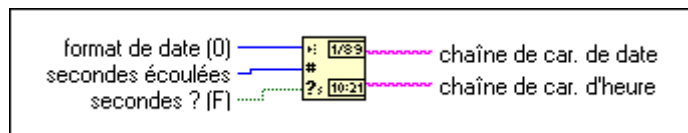
Obtenir la date et l'heure en secondes

Retourne un nombre indépendant de la zone horaire représentant le nombre de secondes écoulées depuis le vendredi 1^{er} janvier 1904 à 00:00, heure universelle.



Obtenir la chaîne de caractères de la date/heure

Convertit un nombre indépendant de la zone horaire représentant le nombre de secondes écoulées depuis le vendredi 1^{er} janvier 1904 à 00:00, heure universelle en chaîne de caractères de la date et de l'heure dans la zone horaire configurée pour votre ordinateur.



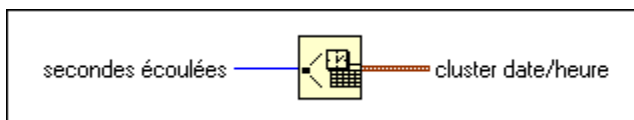
Boîte de dialogue un bouton

Affiche une boîte de dialogue qui contient un message et un seul bouton. Le **nom du bouton** représente le nom affiché sur le bouton de la boîte de dialogue.



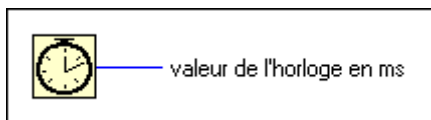
Secondes en date/heure

Convertit en nombre indépendant de la zone horaire représentant le nombre de secondes écoulées depuis le vendredi 1^{er} janvier 1904 à 00:00, heure universelle, en un cluster de neuf entiers 32 bits signés spécifiant l'heure locale (seconde, minute, heure, jour du mois, mois, année, jour de la semaine, jour de l'année et heure standard ou heure d'été) dans la zone horaire configurée pour votre ordinateur. Le paramètre de l'heure d'été est défini selon la configuration d'heure d'été de votre système d'exploitation et indique si le cluster de la date/heure est ajusté sur l'heure d'été.



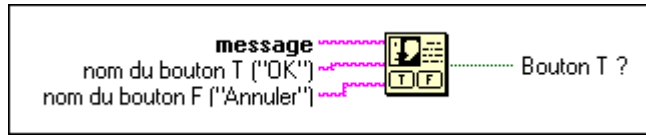
Compteur d'impulsions d'horloge (ms)

Retourne la valeur de l'horloge en millisecondes. L'heure de référence de base (zéro milliseconde) n'est pas définie ; autrement dit, vous ne pouvez pas convertir la **valeur de l'horloge en millisecondes** en une heure ou une date en temps réel. Faites attention quand vous utilisez cette fonction dans des comparaisons car la valeur de l'horloge en millisecondes passe automatiquement de $2^{32}-1$ à 0.



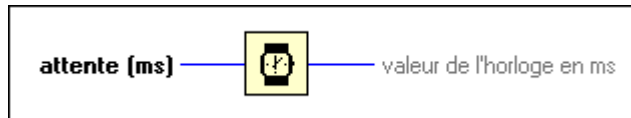
Boîte de dialogue deux boutons

Affiche une boîte de dialogue qui contient un **message** et deux boutons. Les noms affichés sur les boutons de la boîte de dialogue sont **nom du bouton T** et **nom du bouton F**.



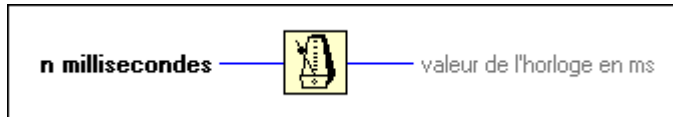
Attendre (ms)

Attend le nombre spécifié de millisecondes et retourne ensuite la valeur de l'horloge en millisecondes.



Attendre un multiple de ms

Attend jusqu'à ce que la valeur de l'horloge en millisecondes devienne un multiple de la valeur **m de millisecondes**. Utilisez cette fonction pour synchroniser des opérations. Vous pouvez aussi invoquer cette fonction dans une boucle pour contrôler la fréquence d'exécution de la boucle. Il est possible néanmoins que la durée de la première boucle soit plus courte.

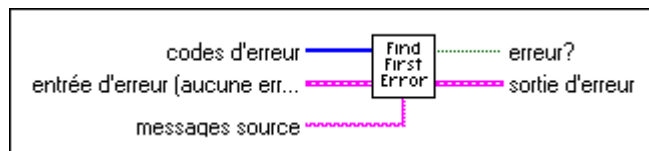


Descriptions des VIs de gestion d'erreur

Les VIs de gestion d'erreur suivants sont disponibles.

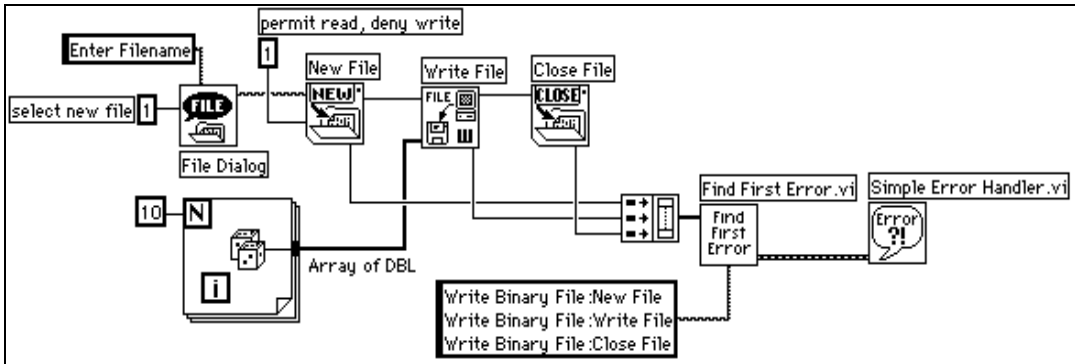
Trouver la première erreur (Find First Error.vi)

Teste l'état d'erreur d'une ou de plusieurs fonctions de bas niveau ou sous-VI produisant un code numérique d'erreur comme sortie.



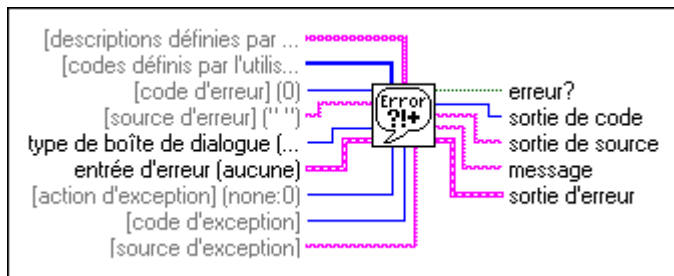
Si ce VI trouve une erreur, il définit les paramètres du cluster **sortie d'erreur**. Vous pouvez câbler ce cluster au Gestionnaire d'erreurs général ou simple pour identifier l'erreur et la signaler à l'utilisateur.

L'illustration ci-dessous montre comment utiliser "Trouver la première erreur" (Find First Error.vi) dans l'exemple Write Binary File. "Trouver la première erreur" (Find First Error.vi) crée un cluster d'erreur à partir de numéros d'erreur individuels et le Gestionnaire simple d'erreur transmet les erreurs à l'utilisateur.



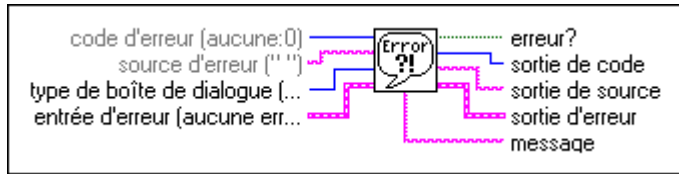
Gestionnaire général d'erreurs

Détermine si une erreur est survenue. Si c'est le cas, ce VI crée une description de l'erreur et peut afficher une boîte de dialogue.



Gestionnaire simple d'erreurs

Détermine si une erreur est survenue. Si c'est le cas, ce VI crée une description de l'erreur et peut afficher une boîte de dialogue.

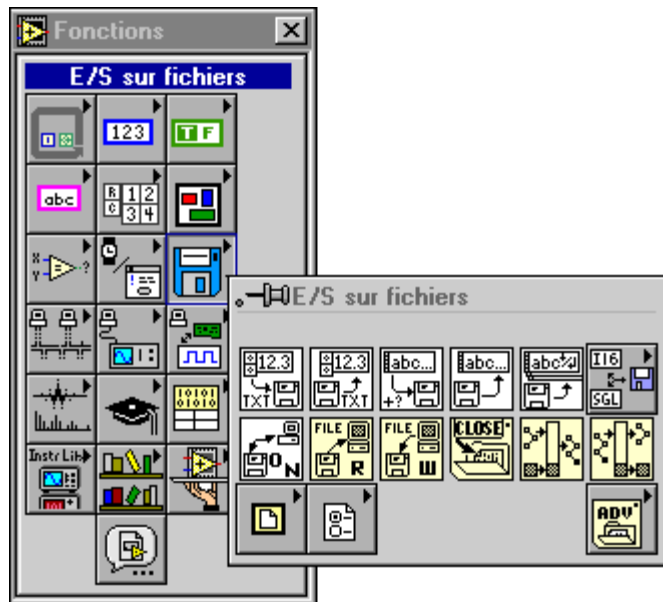


Le Gestionnaire simple d'erreurs appelle le Gestionnaire général d'erreurs et contient les mêmes fonctionnalités de base mais moins d'options.

Fonctions des fichiers

Ce chapitre décrit les VIs de bas niveau et les fonctions utilisés pour manipuler les fichiers, les répertoires et les chemins. Ce chapitre décrit également les constantes de fichiers et les VIs de fichier de niveau supérieur.

Vous pouvez accéder à ces fonctions, constantes et VIs en sélectionnant **Fonctions»E/S sur fichiers**.



La palette **E/S sur fichiers** contient les sous-palettes suivantes :

- Fonctions de fichiers avancées
- VIs de fichiers binaires
- VIs de fichiers de configuration
- Constantes de fichiers

Pour des exemples de fonctions et de VIs de fichiers, reportez-vous à `exemples\file`.

Vue d'ensemble des fonctions et des VIs d'E/S sur fichiers

Cette section présente les VIs de bas et de haut niveaux ainsi que les fonctions de fichiers.

Les VIs de fichiers de haut niveau

Vous pouvez utiliser les VIs de fichiers de haut niveau pour écrire ou lire les types de données suivants :

- Des chaînes de caractères en fichiers texte
- Des tableaux unidimensionnels (1D) ou bidimensionnels (2D) de nombres simple précision en fichiers texte au format tableur
- Des tableaux 1D ou 2D de nombres simple précision ou d'entiers mot signés en fichiers de structure d'octet

Les VIs de fichiers de haut niveau décrits dans ce document appellent les fonctions de fichiers de bas niveau pour exécuter des opérations de fichier complètes et simples à utiliser. Ces VIs permettent d'ouvrir ou de créer des fichiers, d'y écrire ou d'y lire des données et de les fermer. Si une erreur survient, ces VIs affichent une boîte de dialogue qui décrit le problème et vous donne l'option d'interrompre ou de continuer l'exécution.

Les VIs de fichiers de haut niveau se trouvent dans la rangée supérieure de la palette et sont constitués des VIs suivants :

- VIs de fichiers binaires (dans la sous-palette)
- “Lire des caractères dans un fichier” (Read Characters from File.vi)
- “Lire dans un fichier tableur” (Read from Spreadsheet File.vi)
- “Lire des lignes dans un fichier” (Read Lines from File.vi)
- “Ecrire des caractères dans un fichier” (Write Characters to File.vi)
- “Ecrire dans un fichier tableur” (Write to Spreadsheet File.vi)

VIs de bas niveau et fonctions de fichiers

Les VIs de bas niveau et les fonctions de fichiers exécutent une opération de fichier à la fois. Ces VIs et fonctions effectuent une détection d'erreurs en plus de leurs autres fonctions. Les fonctions et les VIs de bas niveau les plus courants se trouvent dans la seconde rangée de la palette. Les fonctions de bas niveau restantes se trouvent dans la sous-palette **Fonctions de fichiers avancées**.

Les principales opérations de fichier de bas niveau impliquent un processus en trois étapes. D'abord, vous devez ouvrir ou créer un fichier. Vous pouvez ensuite écrire des données dans le fichier ou lire des données du fichier. Fermez le fichier pour conclure. D'autres opérations de fichier comprennent : créer des répertoires ; déplacer, copier ou effacer des fichiers ; mettre à jour des fichiers ; lister le contenu de répertoires ; changer les caractéristiques de fichiers et manipuler des chemins.

Quand vous ouvrez ou créez un fichier, vous devez spécifier son emplacement. Différents ordinateurs décrivent l'emplacement des fichiers de manières différentes, mais la plupart des systèmes informatiques utilisent un système hiérarchique pour spécifier l'emplacement des fichiers. Dans un système hiérarchique de fichiers, le système informatique superpose une hiérarchie dans le support de stockage. Vous pouvez stocker des fichiers dans des répertoires, qui peuvent à leur tour contenir d'autres répertoires.

Quand vous spécifiez un fichier ou un répertoire dans un système hiérarchique de fichiers, vous devez indiquer le nom du fichier ou du répertoire, ainsi que son emplacement dans la hiérarchie. De plus, quelques systèmes de fichier supportent la connexion de plusieurs supports discrets, nommés volumes. Par exemple, les systèmes Windows supportent des lecteurs multiples connectés à un système ; pour la plupart de ces systèmes, vous devez inclure le nom du volume pour créer une spécification complète de l'emplacement d'un fichier. Sur d'autres systèmes, comme UNIX, il n'est pas nécessaire de spécifier les emplacements des supports de stockage pour les fichiers car le système d'exploitation vous cache l'implémentation physique du fichier système.

La méthode pour identifier la destination d'une fonction de fichier dépend de si la destination est ou n'est pas un fichier ouvert. Si la destination n'est pas un fichier ouvert, ou si c'est un répertoire, vous devez spécifier une destination en utilisant le *chemin* de la destination. Le chemin décrit le volume contenant la destination, les répertoires entre le haut niveau et la destination, et le nom de la destination. Si la destination est un fichier ouvert, utilisez un *refnum de fichier* pour identifier le fichier à manipuler. Le refnum de fichier est un identificateur associé au fichier lors de son ouverture. Quand vous fermez le fichier, le gestionnaire de fichiers sépare le refnum de fichier du fichier. Autrement dit, le refnum est périmé quand le fichier est fermé.

Reportez-vous au *LabVIEW Online Tutorial: Introduction to LabVIEW* pour plus d'informations sur la spécification de chemins en G et pour des exemples de fonctions de fichier.

Fichiers de structure d'octets et d'enregistrement de données

Le G peut créer et donner accès à deux types de fichiers : des fichiers de structure d'octets et d'enregistrement de données.

Un fichier *de structure d'octets*, comme son nom l'indique, est un fichier dont l'unité de base est un octet. Un fichier de structure d'octets peut contenir un ensemble homogène d'un type de données G ou encore une collection arbitraire de types de données (caractères, nombres, booléens, tableaux, chaînes de caractères, clusters, etc.). Un fichier de texte ASCII, contenant ce paragraphe, est l'exemple le plus simple de fichier de structure d'octets. Un fichier de structure d'octets peut aussi être un fichier texte tableur, qui comporte des rangées de nombres ASCII, où les nombres sont séparés par des tabulations et les rangées séparées par des retours chariot.

Un autre exemple simple de fichier de structure d'octets est un tableau d'entiers binaires 16 bits ou de nombres simple précision à virgule flottante, obtenus grâce à un programme d'acquisition de données (DAQ). Un fichier de structure d'octets plus complexe peut comporter un tableau d'entiers 16 bits ou de nombres simple précision à virgule flottante, précédé par un en-tête en texte ASCII qui décrit comment et quand ont été obtenues les données. Cet en-tête pourrait être un cluster de paramètres d'acquisition, tels qu'un tableau de canaux et de facteurs d'échelle, une vitesse de balayage et ainsi de suite.

Un fichier de travail Excel, par opposition à un fichier texte Excel, est aussi un exemple de forme plus complexe de fichier de structure d'octets car il contient du texte auquel sont associées des données de formatage spécifiques à Excel qui n'ont aucun sens quand vous les lisez comme du texte. En bref, vous pouvez créer un fichier de structure d'octets contenant chaque type de données G. Les fichiers de structure d'octets peuvent être créés avec des VIs de fichiers de haut niveau et de bas niveau ainsi qu'avec des fonctions.

Un fichier *d'enregistrement de données*, quant à lui, contient une séquence d'enregistrements dont la structure est identique. Tout comme les fichiers de structure d'octets, les composantes d'un enregistrement de données peuvent être de n'importe quel type de données G. La différence est que tous les enregistrements de données doivent être du même type. Les fichiers d'enregistrement de données ne peuvent être créés qu'avec des fonctions de fichier de bas niveau.

En général, vous pouvez écrire dans un fichier de structure d'octets en assemblant des chaînes de caractères, des nombres ou des tableaux de nombres de n'importe quelle longueur. Vous pouvez également écraser des données n'importe où dans le fichier. Vous pouvez écrire dans un fichier d'enregistrement de données en ajoutant un enregistrement à la fois. Vous ne pouvez cependant pas écraser l'enregistrement.

Vous pouvez lire un fichier de structure d'octets en spécifiant l'offset ou l'indice d'octets et le nombre d'instances du type de structure d'octets spécifié que vous souhaitez lire. Vous pouvez lire un fichier d'enregistrement de données en spécifiant l'offset ou l'indice d'enregistrement et le nombre d'enregistrements que vous souhaitez lire.

En général, vous devez utiliser des fichiers de structure d'octets pour des données texte ou tableur devant être lues par d'autres applications. Vous pouvez utiliser des fichiers de structure d'octets pour enregistrer des données continuellement acquises que vous devez lire séquentiellement ou aléatoirement en quantités arbitraires. En général, vous devez utiliser des fichiers d'enregistrement de données pour enregistrer plusieurs résultats de tests ou de signaux que vous pouvez lire un par un et que vous traitez individuellement. Les fichiers d'enregistrement de données sont difficiles à lire à partir d'applications non G.

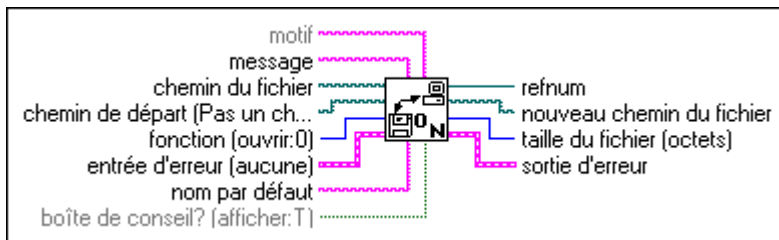
Paramètres transmissibles

Plusieurs fonctions de fichiers comportent des *paramètres transmissibles*, qui retournent la même valeur que le paramètre d'entrée. Vous pouvez utiliser ces paramètres pour contrôler l'ordre d'exécution des fonctions. En reliant la sortie transmissible du premier nœud que vous souhaitez exécuter à l'entrée correspondante du nœud suivant, vous créez une dépendance artificielle des données. Sans ces paramètres transmissibles, vous auriez souvent recours aux structures Séquence pour assurer que les opérations de fichiers d'E/S ont lieu dans le bon ordre.

E/S d'erreurs dans les fonctions d'E/S sur fichiers

Le G utilise des clusters d'E/S d'erreur, dont **entrée d'erreur** et **sortie d'erreur**, dans toutes ses fonctions d'E/S sur fichier. Avec des clusters d'E/S d'erreur, vous pouvez enchaîner plusieurs fonctions. Quand une erreur survient dans une fonction, la fonction la transmet à la fonction suivante. Quand une erreur est transmise à une fonction, celle-ci ne s'exécute pas et transmet l'erreur à la fonction suivante et ainsi de suite.

L'illustration suivante montre un exemple de clusters **entrée d'erreur** et **sortie d'erreur**.



Bien que les clusters d'E/S d'erreur spécifient si une erreur est survenue, vous pouvez utiliser des gestionnaires d'erreurs pour avertir l'utilisateur qu'une erreur s'est produite. Pour plus d'informations sur l'E/S d'erreurs, reportez-vous au chapitre 10, *Fonctions de temps, de dialogue et d'erreurs*, dans ce manuel.

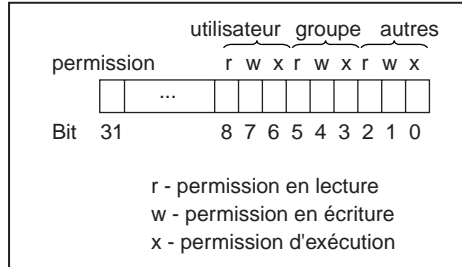
Autorisations

Quelques fonctions de fichiers disposent d'un paramètre entier 32 bits nommé **autorisations** ou **nouvelles autorisations**. Ces fonctions n'utilisent que les neuf bits de moindre poids de l'entier 32 bits pour déterminer les autorisations d'accès aux fichiers et aux répertoires.

(Windows) Les autorisations sont ignorées pour les répertoires. Pour les fichiers, seul le bit 7 (le bit d'autorisation d'écriture UNIX) est utilisé. Si ce bit est libre, le fichier est en lecture seule. Sinon, vous pouvez écrire dans le fichier.

(Macintosh) Les 9 bits d'autorisations sont utilisés pour les répertoires. Les bits qui contrôlent les autorisations de lecture, écriture et exécutions, respectivement, sur un système UNIX, sont utilisés pour contrôler les droits d'accès See File, Make Changers et See Folder, respectivement, sur le Macintosh. Pour les fichiers, seul le bit 7 (le bit d'autorisation d'écriture UNIX) est utilisé. Si ce bit est libre, le fichier est verrouillé, sinon le fichier ne l'est pas.

(UNIX) Les 9 bits des autorisations correspondent exactement aux 9 bits d'autorisation UNIX qui gouvernent les autorisations de lecture, écriture et des exécutions pour les utilisateurs et les groupes, entre autres. L'illustration suivante montre les bits d'autorisation sur un système UNIX.

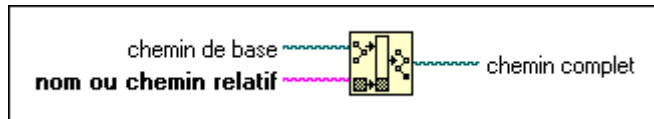


Descriptions des VIs et des fonctions d'E/S sur fichiers

Les fonctions et les VIs suivantes sont disponibles dans la palette **E/S sur fichiers**.

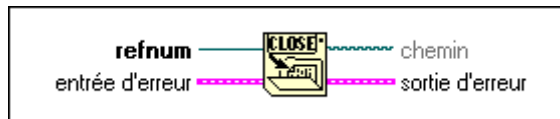
Construire un chemin

Construit un nouveau chemin en ajoutant un nom (ou chemin relatif) à un chemin existant.



Fermer un fichier

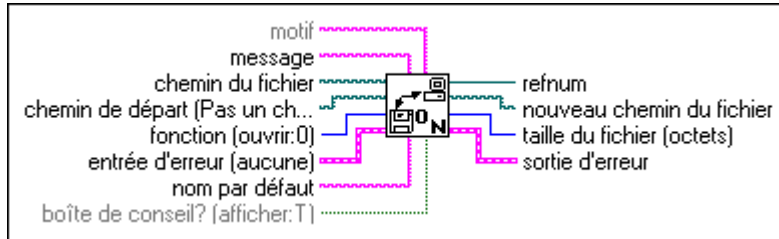
Ecrit tous les buffers du fichier identifiés par **refnum** sur le disque, met à jour l'entrée répertoire du fichier, ferme le fichier et annule **refnum** pour des opérations de fichier ultérieures.



Remarque Cette fonction manipule les erreurs différemment des autres fonctions ; elle s'exécute même quand entrée d'erreur indique qu'une erreur est survenue lors de l'opération précédente.

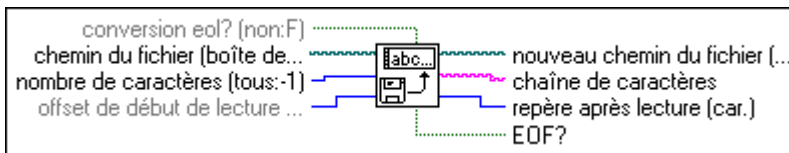
Ouvrir/Créer/Remplacer un fichier (Open/Create/Replace File.vi)

Ouvre un fichier existant, crée un nouveau fichier ou remplace un fichier existant, par programme ou de manière interactive en utilisant une boîte de dialogue de fichier. Vous pouvez choisir de spécifier un **message** de dialogue, un nom de fichier par défaut, un **chemin de départ** ou un **motif** de fichier. Utilisez ce VI avec les fonctions Ecrire ou “Lire un fichier”.



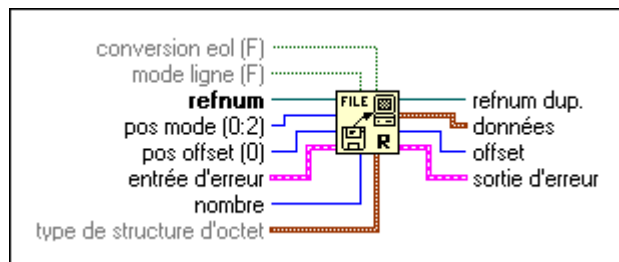
Lire des caractères dans un fichier (Read Characters from File.vi)

Lit un nombre spécifié de caractères dans un fichier de structure d'octets à partir d'un offset de caractère spécifié. Le VI ouvre le fichier avant de le lire et le ferme ensuite.



Lire un fichier

Lit les données du fichier spécifié par le **refnum** et retourne les **données**. La lecture commence à l'emplacement spécifié par **pos mode** et **pos offset** et dépend du format du fichier spécifié.



Lecture de fichiers de structure d'octets

Si **refnum** correspond à un fichier de structure d'octets, la fonction “Lire un fichier” lit les données du fichier de structure d'octets spécifié par **refnum**. Vous pouvez câbler soit le **mode ligne** soit le **type de structure d'octets** quand vous pouvez lire des fichiers de structure d'octets, mais vous ne pouvez pas câbler les deux. Si vous ne câblez pas le **type de structure d'octets**, “Lire un fichier” considère que les données qui commencent à l'offset d'octets désigné sont une chaîne de caractères. Si vous câblez le **type de structure d'octets**, la fonction interprète les **données** qui commencent à l'offset d'octets désigné comme **nombre** d'instances de ce type. Après l'opération de lecture, la fonction définit le repère du fichier sur l'octet qui suit le dernier octet lu. Si la fonction se trouve à la fin du fichier avant d'avoir lu toutes les données requises, elle retourne autant d'instances complètes du **type de structure d'octets** désignée que possible.

Lecture de caractères

Pour lire des caractères dans un fichier de structure d'octets (normalement un fichier texte), ne câblez pas le **type de structure d'octets**. Les paragraphes suivants décrivent comment fonctionnent les paramètres **mode ligne**, **nombre**, **conversion eol** et **données** pendant la lecture d'un fichier de structure d'octets.

Les paramètres **mode ligne** et **nombre** déterminent quand s'arrête la lecture.

Si le **mode ligne** est Vrai (TRUE) et que vous ne câblez pas **nombre** ou si **nombre** est égal à 0, “Lire un fichier” lit jusqu'au repère de fin de ligne (un retour chariot, un retour à la ligne ou un retour chariot suivi d'un retour à la ligne), ou jusqu'à la fin du fichier. Si le **mode ligne** est Vrai (TRUE) et que **nombre** est supérieur à 0, “Lire un fichier” lit jusqu'au repère de fin de ligne, jusqu'à la fin du fichier, ou bien lit **nombre** caractères.

Si le **mode ligne** est Faux (FALSE), “Lire un fichier” lit **nombre** caractères. Dans ce cas, si vous ne câblez pas le paramètre **nombre**, sa valeur par défaut est 0. La valeur par défaut du **mode ligne** est Faux (FALSE).

Conversion eol (F) détermine si la fonction convertit les repères de fin de ligne lus en repères de fin de ligne G. Le repère de fin de ligne spécifique au système est un retour chariot suivi d'un retour à la ligne sous Windows, un retour chariot sous Macintosh et un retour à la ligne sous UNIX. Le repère de fin de ligne G est un retour à la ligne.

Si **conversion eol** est Vrai (TRUE), la fonction convertit tous les repères de fin de ligne trouvés en retours à la ligne. Si **conversion eol** est Faux (FALSE), la fonction ne convertit pas les repères de fin de ligne lus. La valeur par défaut de **conversion eol** est Faux (FALSE).

Les **données** correspondent à la chaîne de caractères lue dans le fichier.

Lecture de données binaires

Pour lire les données binaires d'un fichier de structure d'octets, câblez le type de données sur **type de structure d'octets**. Dans ce cas, **nombre** et **données** fonctionnent comme décrit dans les paragraphes suivants et il n'est pas nécessaire de câbler **mode ligne** ou **conversion eol**.

Le **type de structure d'octets** peut être n'importe quel type de données. "Lire un fichier" interprète les données en commençant à l'offset d'octets désigné comme **nombre** d'instances de ce type. Si le type est de longueur variable, c'est-à-dire, un tableau, une chaîne de caractères, ou un cluster contenant un tableau ou une chaîne de caractères, la fonction suppose que chaque instance de ce type comporte la longueur ou les dimensions de cette instance. Si ce n'est pas le cas, la fonction interprète incorrectement les données. Si "Lire un fichier" détermine que les données ne correspondent pas au type, la fonction retourne des **données** égales à la valeur par défaut pour son type et retourne une erreur.

Le **nombre** représente le nombre d'instances du **type de structure d'octets** à lire. Si **nombre** n'est pas câblé, la fonction retourne une seule instance du **type de structure d'octets**.

Si vous câblez le **nombre**, il peut être un nombre scalaire, et dans ce cas, la fonction retourne un tableau 1-D d'instances du **type de structure d'octets**. Ou il peut être un cluster de N nombres scalaires, et dans ce cas, la fonction retourne un tableau de N dimensions d'instances du **type de structure d'octets**.

Si le **nombre** câblé est un nombre scalaire et que le **type de structure d'octets** n'est pas un tableau, la fonction retourne ce nombre d'instances dans un tableau 1D. Par exemple, si le type est un nombre simple précision à virgule flottante, et que le **nombre** est 3, la fonction retourne un tableau de trois nombres simple précision à virgule flottante. Cependant, si le type est un tableau, la fonction retourne les instances dans un tableau de clusters, car le G ne supporte pas des tableaux de tableaux. Ainsi, si le type est un tableau de nombres simple précision à virgule flottante, et si le **nombre** est 3, la fonction retourne un tableau de clusters de trois tableaux de nombres simple précision à virgule flottante.

Si le **nombre** câblé est un cluster de N nombres, la fonction retourne un tableau de N dimensions des instances du type. La taille de chaque dimension représente la valeur du nombre correspondant selon l'ordre du cluster. Le nombre d'instances retournées de cette manière est le produit des N nombres. Ainsi, vous pouvez retourner 20 nombres simple précision à virgule flottante, comme un tableau 2D de deux colonnes et 10 rangées en câblant un cluster de deux éléments contenant l'élément 0 = 2 et l'élément 1 = 10 à l'entrée **nombre**.

Les **données** contiennent les données lues dans le fichier. Reportez-vous à la description précédente du paramètre **nombre** pour une explication des structures que les données peuvent avoir.

Lecture de fichiers d'enregistrement de données

Si **refnum** correspond à un fichier d'enregistrement de données, la fonction "Lire un fichier" lit les enregistrements du fichier d'enregistrement de données spécifié par le **refnum**. Si les données du fichier ne correspondent pas au type de données associé au fichier d'enregistrement de données, cette fonction retourne une erreur.

Le nombre d'enregistrements lus peut être inférieur à celui spécifié par **nombre** si la fonction atteint la fin du fichier. La fonction définit le repère de fichier sur l'enregistrement qui suit le dernier enregistrement lu. (Il ne devrait jamais y avoir d'enregistrement incomplet ; sinon, le fichier est corrompu.)

Ne câblez pas **conversion eol**, **mode ligne** et **type de structure d'octets**. Ils ne sont pas appropriés pour les fichiers d'enregistrement de données. Les paramètres **nombre** et **données** fonctionnent de la manière suivante.

Le **nombre** représente le nombre d'enregistrements à lire et peut être câblé ou non. Si vous ne câblez pas le **nombre**, la fonction retourne un seul enregistrement du type d'enregistrement de données spécifié quand le fichier est créé ou ouvert. Par exemple, si le type est un entier 16 bits, la fonction retourne un entier 16 bits. Si le type est un tableau d'entiers 16 bits, la fonction retourne un tableau d'entiers 16 bits. (Vos enregistrements comprennent généralement des clusters d'éléments divers, mais les règles pour les types simples utilisés dans ces exemples s'appliquent également.)

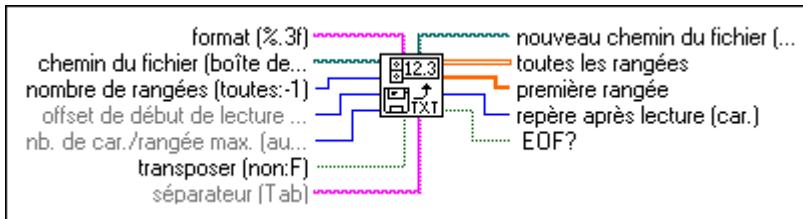
Si vous câblez le **nombre**, il peut être un nombre scalaire, dans ce cas, la fonction retourne un tableau 1D d'enregistrements. Ou il peut être un cluster de N nombres scalaires, dans ce cas, la fonction retourne un tableau de N dimensions d'enregistrements.

Si le **nombre** câblé est un nombre scalaire et si le type d'enregistrement de données n'est pas un tableau, la fonction retourne le nombre d'enregistrements dans un tableau 1D. Par exemple, si le type est un nombre simple précision à virgule flottante, et que le **nombre** est 3, le tableau contient trois nombres simple précision à virgule flottante. Cependant, si le type est un tableau, la fonction retourne les enregistrements dans un tableau de clusters, car le G ne supporte pas des tableaux de tableaux. Ainsi, si le type est un tableau de nombres simple précision à virgule flottante, et si le **nombre** est 3, la fonction retourne un tableau de clusters de trois tableaux de nombres simple précision à virgule flottante.

Si le **nombre** câblé est un cluster de N nombres, la fonction retourne un tableau d'enregistrement de N dimensions. La taille de chaque dimension représente la valeur du nombre correspondant selon l'ordre du cluster. Le nombre d'enregistrements retourné de cette manière est égal au produit des N nombres. Ainsi, vous pouvez retourner 20 nombres simple précision à virgule flottante, comme un tableau 2D de deux colonnes et 10 rangées en câblant un cluster de deux éléments contenant l'élément 0 = 2 et l'élément 1 = 10 à l'entrée **nombre**.

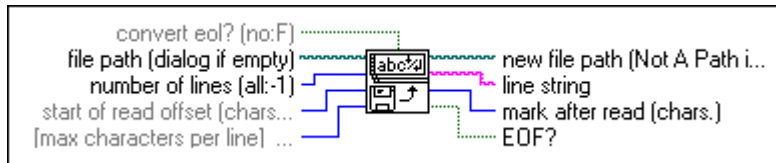
Lire dans un fichier tableur

Lit un nombre spécifié de lignes ou de rangées dans un fichier texte numérique à partir d'un caractère offset spécifié et convertit les données en tableau 2D de nombres simple précision. Vous pouvez choisir de transposer le tableau. Le VI ouvre le fichier avant sa lecture et le ferme ensuite. Vous pouvez utiliser ce VI pour lire un fichier tableur sauvegardé au format texte. Ce VI invoque la fonction "Chaîne de caractères au format tableur en tableau" pour convertir les données.



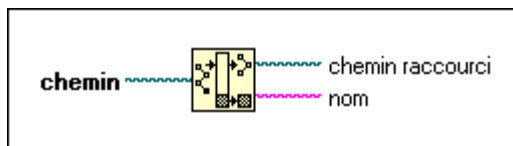
Lire des lignes dans un fichier

Lit un nombre spécifié de lignes dans un fichier de structure d'octets à partir d'un caractère offset spécifié. Le VI ouvre le fichier avant de le lire et le ferme après.



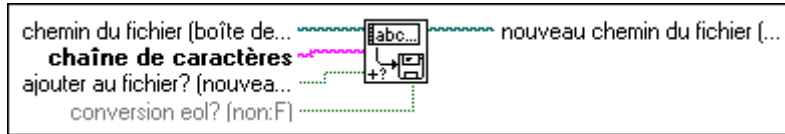
Dépouiller un chemin

Retourne le nom de la dernière composante d'un chemin et le **chemin dépouillé** qui mène à cette composante.



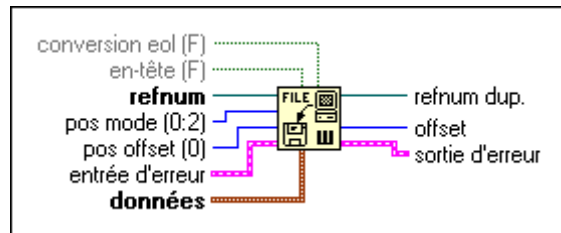
Ecrire des caractères dans un fichier

Ecrit une chaîne de caractères dans un nouveau fichier de structure d'octets ou ajoute la chaîne de caractères dans un fichier existant. Le VI ouvre ou crée le fichier avant d'y écrire et le ferme ensuite.



Ecrire dans un fichier

Ecrit des données dans le fichier spécifié par **refnum**. L'écriture commence à l'emplacement spécifié par **pos mode** et **pos offset** pour le fichier de structure d'octets et à la fin du fichier pour les fichiers d'enregistrement de données. Les **données**, l'**en-tête** et le format du fichier spécifiés déterminent le volume de données écrites.



Ecrire des fichiers de structure d'octets

Si **refnum** correspond à un fichier de structure d'octets, la fonction "Ecrire dans un fichier" écrit dans un emplacement spécifié par **pos mode** et **pos offset** dans le fichier de structure d'octets spécifié par **refnum**. Si le type de données de haut niveau est de longueur variable (c'est-à-dire, une chaîne de caractères ou un tableau), "Ecrire dans un fichier" peut écrire un **en-tête** dans le fichier qui spécifie la taille des données. "Ecrire dans un fichier" définit le repère de fichier sur l'octet qui suit le dernier octet écrit. **Conversion eol** détermine si la fonction convertit les repères de fin de ligne écrits en repères de fin de ligne spécifiques au système. Vous ne pouvez câbler **conversion eol** que si les **données** sont une chaîne de caractères. Le repère de fin de ligne spécifique au système est un retour chariot suivi d'un retour à la ligne sous Windows, un retour à la ligne sous UNIX et un retour chariot sous Macintosh. Si **en-tête** est Vrai (TRUE), "Ecrire dans un fichier" ignore le paramètre **conversion eol**.

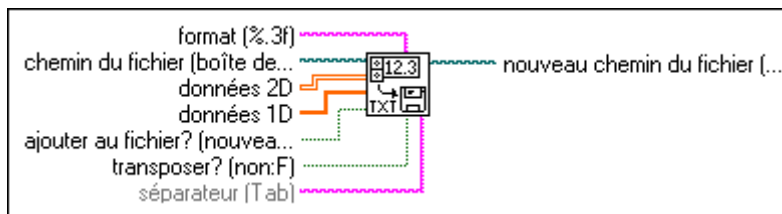
Ecrire des fichiers d'enregistrement de données

Si **refnum** correspond à un fichier d'enregistrement de données, la fonction "Ecrire dans un fichier" écrit des données en tant qu'enregistrements dans le fichier d'enregistrement de données spécifié par **refnum**. L'écriture commence toujours à la fin du fichier d'enregistrement de données (les fichiers d'enregistrement de données sont uniquement des fichiers d'ajout). "Ecrire dans un fichier" définit le repère de fichier sur l'enregistrement qui suit le dernier enregistrement écrit. Les paramètres **conversion eol**, **en-tête**, **pos mode** et **pos offset** ne s'appliquent pas aux fichiers d'enregistrement de données et vous ne pouvez pas les câbler. Le paramètre **données** fonctionne de la façon suivante pour les fichiers d'enregistrement de données.

Les **données** doivent correspondre au type de données spécifié quand vous ouvrez ou créez le fichier ou doivent être un tableau du même type de données. Pour le premier type, cette fonction écrit des **données** en tant qu'enregistrement unique dans le fichier d'enregistrement de données. La représentation de données numériques est contrainte dans la représentation du type de données si nécessaire. Pour le second type, cette fonction écrit chaque élément de données en tant qu'enregistrement séparé dans le fichier d'enregistrement de données en commençant par la rangée principale.

Ecrire dans un fichier tableur

Convertit un tableau 2D ou 1D de nombres simple précision en chaîne textuelle et écrit la chaîne de caractères dans un nouveau fichier de structure d'octets ou ajoute la chaîne de caractères dans un fichier existant. Vous pouvez choisir de transposer les données. Ce VI ouvre ou crée le fichier avant d'y écrire et le ferme ensuite. Vous pouvez utiliser ce VI pour créer un fichier texte pouvant être lu par la plupart des applications tableurs. Ce VI appelle la fonction "Tableau en chaîne de caractères au format tableur" pour convertir les données.



Descriptions des VIs de fichiers binaires

Les VIs suivants sont disponibles dans la sous-palette **VIs de fichiers binaires**.



Lire dans un fichier I16 (Read from I16 File.vi)

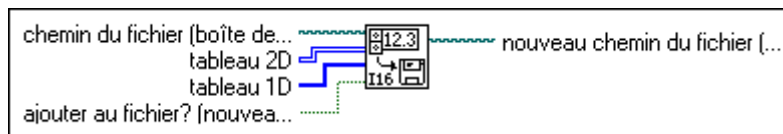
Lit un tableau de données 2D ou 1D dans un fichier de structure d'octets d'entiers mot signés (I16). Le VI ouvre le fichier avant de le lire et le ferme ensuite. Vous pouvez utiliser ce VI pour lire des données binaires ou non mises à l'échelle acquises par le biais de VIs d'acquisition de données et écrites dans un fichier avec le VI "Ecrire dans un fichier I16" (Write to I16 File.vi).

Lire dans un fichier SGL (Read from SGL File.vi)

Lit un tableau de données 2D ou 1D dans un fichier de structure d'octets de nombres simple précision. Le VI ouvre le fichier avant la lecture et le ferme ensuite. Vous pouvez utiliser ce VI pour lire des données mises à l'échelle obtenues par le biais de VIs d'acquisition de données et écrites dans un fichier avec le VI "Ecrire dans un fichier SGL" (Write to SGL File.vi).

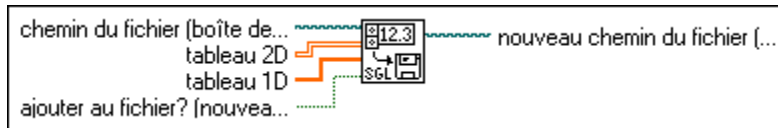
Ecrire dans un fichier I16 (Write to I16 File.vi)

Ecrit un tableau 2D ou 1D d'entiers mot signés (I16) dans un nouveau fichier de structure d'octets ou ajoute des données à un fichier existant. Le VI ouvre ou crée le fichier avant d'y écrire et le ferme ensuite. Vous pouvez utiliser ce VI pour écrire des données binaires ou non mises à l'échelle fournies par des VIs d'acquisition de données.



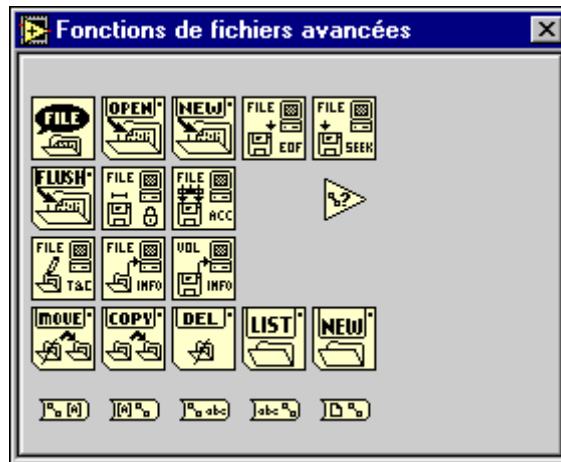
Ecrire dans un fichier SGL (Write to SGL File.vi)

Écrit un tableau 2D ou 1D de nombres simple précision dans un nouveau fichier de structure d'octets ou ajoute des données à un fichier existant. Le VI ouvre ou crée le fichier avant d'y écrire et le ferme ensuite. Vous pouvez utiliser ce VI pour écrire des données mises à l'échelle fournies par des VIs d'acquisition sans changer la représentation.



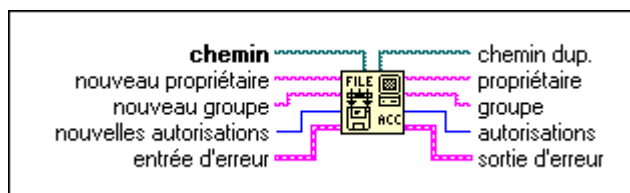
Descriptions des fonctions de fichiers avancées

Les fonctions suivantes sont disponibles dans la sous-palette **Fonctions de fichiers avancées**.



Droits d'accès

Définit et retourne le propriétaire, le groupe et les autorisations du fichier ou du répertoire spécifié par **chemin**. Si vous ne spécifiez pas un **nouveau propriétaire**, un **nouveau groupe**, ou de **nouvelles autorisations**, cette fonction retourne les paramètres courants non modifiés.

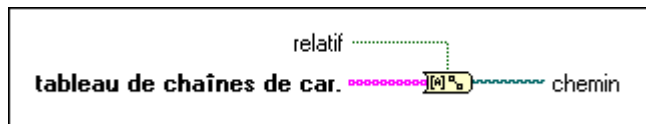


(Windows) La fonction “Droits d’accès” ignore les paramètres **nouveau propriétaire** et **nouveau groupe** et retourne des chaînes de caractères vides pour **propriétaire** et **groupe** car Windows ne supporte pas les propriétaires et les groupes.

(Macintosh) Si **chemin** se rapporte dans un fichier, la fonction “Droits d’accès” ignore **nouveau propriétaire** et **nouveau groupe** et retourne des chaînes de caractères vides pour **propriétaire** et **groupe** car Macintosh ne supporte pas les propriétaires ou les groupes pour les fichiers.

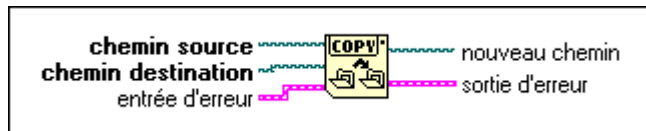
Tableau de chaînes de caractères en chemin

Convertit un **tableau de chaînes de caractères** en un **chemin** relatif ou absolu.



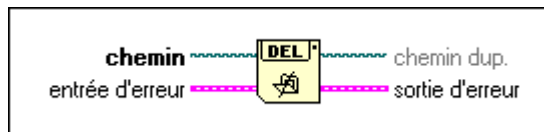
Copier

Copie un fichier ou un répertoire spécifié par **chemin source** dans l’emplacement spécifié par **chemin destination**. Si vous copiez un répertoire, cette fonction copie tout son contenu de manière récursive.



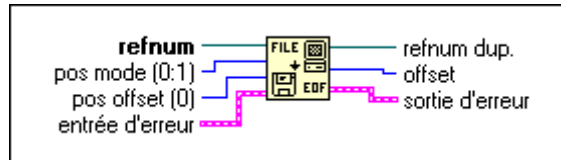
Effacer

Efface un fichier ou un répertoire spécifié par **chemin**. Si **chemin** spécifie un répertoire qui n’est pas vide ou si vous n’avez pas de permission d’écriture pour le fichier ou le répertoire spécifié par **chemin** et son répertoire parent, cette fonction ne supprime pas le répertoire et retourne une erreur.



EOF

Définit et retourne l'EOF (fin-de-fichier) logique du fichier identifié par le **refnum**. **Pos mode** et **pos offset** spécifient le nouvel emplacement d'EOF. Si vous ne spécifiez pas **pos mode** ou **pos offset**, cette fonction retourne l'EOF courant sans modification. Cette fonction retourne toujours l'emplacement de l'EOF par rapport au début du fichier.



Vous ne pouvez pas définir l'EOF d'un fichier d'enregistrement de données. Si **refnum** identifie un fichier d'enregistrement de données, vous ne pouvez pas câbler **pos mode** et **pos offset**. Vous pouvez cependant obtenir l'EOF d'un fichier d'enregistrement de données, qui vous indique combien d'enregistrements se trouvent dans le fichier.

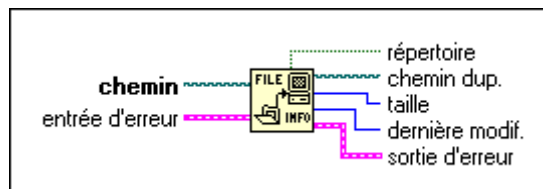
Boîte de dialogue de fichier

Affiche une boîte de dialogue vous permettant de spécifier le chemin d'un fichier ou d'un répertoire. Vous pouvez utiliser cette boîte de dialogue pour sélectionner des fichiers ou répertoires existants ou pour sélectionner l'emplacement et le nom d'un nouveau fichier ou répertoire.



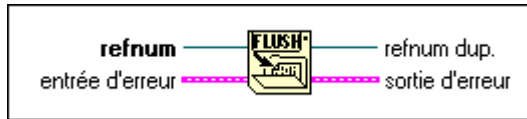
Infos sur le fichier/répertoire

Retourne des informations sur un fichier ou un répertoire spécifié par **chemin**, notamment sa **taille**, la date de sa plus récente modification et si c'est un répertoire.



Mettre à jour le fichier

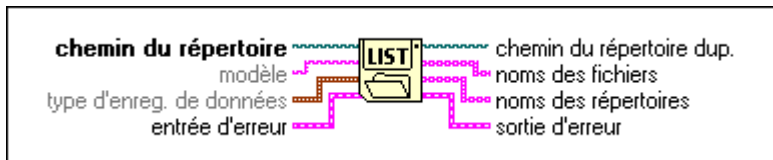
Écrit tous les buffers du fichier identifié par **refnum** sur le disque et met à jour son entrée de répertoire. Le fichier reste ouvert et le **refnum** reste valable.



Les données à écrire dans un fichier sont souvent stockées dans un buffer jusqu'à ce que le buffer soit plein ou que vous fermiez le fichier. Cette fonction force le système d'exploitation à écrire toutes les données du buffer dans le fichier.

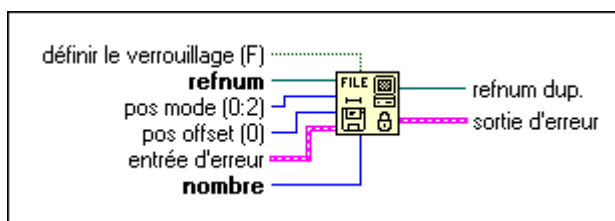
Lister un répertoire

Retourne deux tableaux de chaînes de caractères listant les noms de tous les fichiers et répertoires se trouvant dans le **chemin du répertoire**. La fonction filtre les deux tableaux en se basant sur le **pattern** et filtre le tableau de **noms de fichier** en se basant sur le **type d'enregistrement de données**.



Verrouiller la gamme

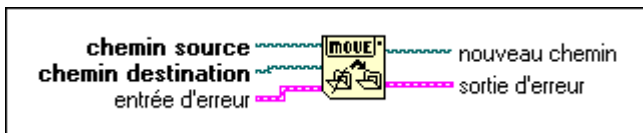
Verrouille ou déverrouille la gamme d'un fichier spécifié par **refnum**. Verrouiller une gamme de fichier empêche la lecture et l'écriture par d'autres utilisateurs, ignorant les autorisations du fichier et le mode d'accès associé au **refnum**. Reportez-vous à la section *Vue d'ensemble des fonctions et des VIs d'E/S sur fichiers* figurant plus haut dans ce chapitre pour des informations détaillées sur les autorisations. Déverrouiller une gamme de fichier supprime l'annulation créée par le verrouillage de la gamme ; les autorisations du fichier et le mode d'accès associé au **refnum** déterminent si d'autres utilisateurs peuvent lire ou écrire dans la gamme du fichier.



Vous ne pouvez pas verrouiller la gamme d'un fichier d'enregistrement de données.

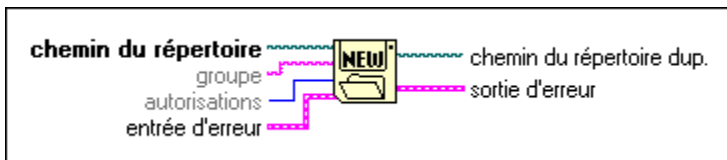
Déplacer

Déplace un fichier ou un répertoire spécifié par **chemin source** vers l'emplacement spécifié par **chemin destination**.



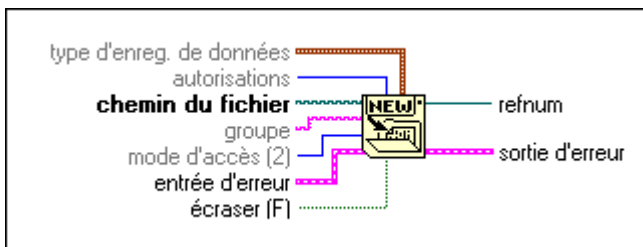
Nouveau répertoire

Crée le répertoire spécifié par **chemin de répertoire**. Si un fichier ou un répertoire existe déjà à l'emplacement spécifié, cette fonction retourne une erreur au lieu d'écraser le fichier ou le répertoire existant.



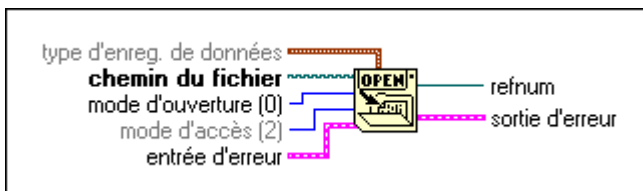
Nouveau fichier

Crée le fichier spécifié par **chemin du fichier** et l'ouvre pour le lire et y écrire (sans tenir compte des **autorisations**).



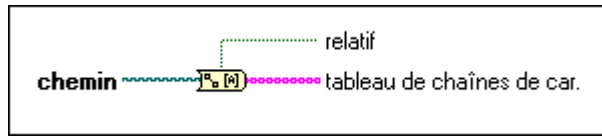
Ouvrir un fichier

Ouvre le fichier spécifié par **chemin du fichier** pour le lire et/ou y écrire.



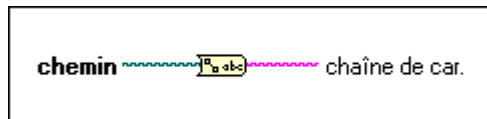
Chemin en tableau de chaînes de caractères

Convertit un **chemin** en **tableau de chaînes de caractères** et indique si le chemin est **relatif**.



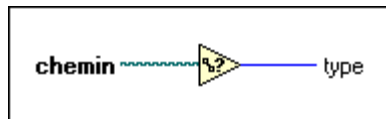
Chemin en chaîne de caractères

Convertit un **chemin** en une chaîne de caractères décrivant le chemin au format standard de la plate-forme.



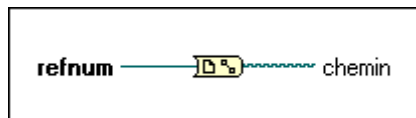
Type de chemin

Retourne le type du chemin spécifié, indiquant si c'est un chemin absolu, relatif ou invalide. Cette fonction ne vérifie que le format du chemin et non si le chemin se rapporte à un fichier ou à un répertoire existant. Ainsi, cette fonction n'indique un chemin invalide que pour la valeur Pas un chemin.



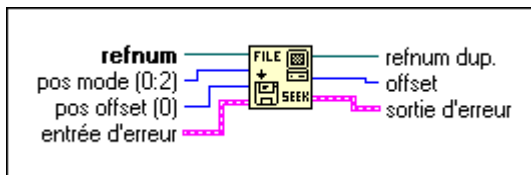
Refnum en chemin

Retourne le chemin associé au **refnum** spécifié.



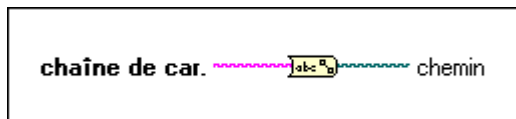
Rechercher

Déplace le repère courant du fichier identifié par le **refnum** à l'emplacement indiqué par **pos offset** selon le mode défini par **pos mode**.



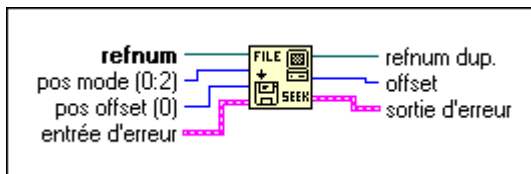
Chaîne de caractères en chemin

Convertit **une chaîne de caractères**, décrivant un chemin au format standard pour la plate-forme courante, en **chemin**.



Type et Créateur

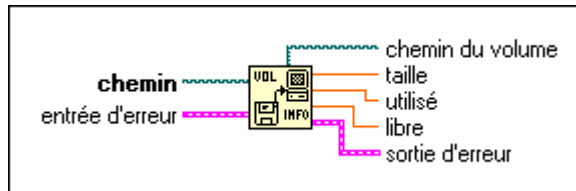
Lit et définit le type et créateur du fichier spécifié par **chemin**. Les types et créateurs de fichier sont des chaînes de quatre caractères. Si vous ne spécifiez pas **nouveau type** ou **nouveau créateur**, cette fonction retourne les paramètres courants sans les modifier.



Windows et UNIX ne supportent pas les types et créateurs de fichiers. Définir le type ou créateur d'un fichier sous ces plates-formes retourne une erreur ; cependant, vous pouvez obtenir le type et créateur de fichier sous ces plates-formes. Si le fichier spécifié comporte un nom se terminant par des caractères reconnus par le type et le créateur comme caractères de spécification d'un type de fichier (.vi pour le type de fichier LVIN et .lib pour le type de fichier LVAR, par exemple), cette fonction retourne ce type dans **type** et LBVW dans **créateur**. Sinon, la fonction retourne ???? dans **type** et dans **créateur**.

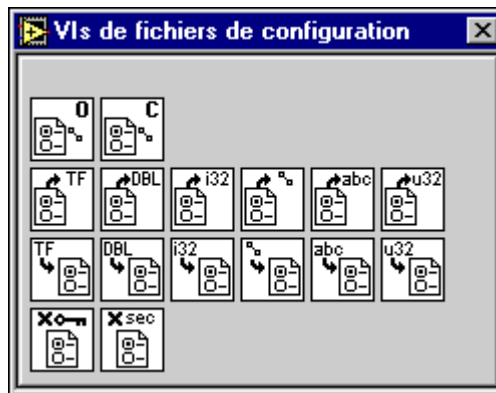
Infos du volume

Retourne des informations sur le volume contenant le fichier ou répertoire spécifié par **chemin**, y compris l'espace total de stockage fourni par le volume, la quantité utilisée et la quantité disponible d'octets.



VI de fichiers de configuration

Les VIs de fichiers de configuration sont des outils utilisés pour créer, modifier et lire des fichiers de configuration indépendants de la plate-forme. L'illustration suivante montre les options disponibles dans la sous-palette **VIs de fichiers de configuration**.



Les VIs de fichiers de configuration opèrent sur un fichier de configuration indépendant de la plate-forme dont le format est comparable au fichier d'initialisation (.ini) standard de Windows.

Le fichier est divisé en sections, libellées par un nom entre crochets. Chaque section doit comporter un nom unique ainsi que des paires de valeurs et de clés. Chaque clé doit comporter un nom unique.

Exemple d'un fichier de configuration contenant une section 1 et une section 2 :

```
[section 1]
key1="string value 1"
key2="string value 2"
key3=53
[section 2]
key1=TRUE
key2=-12.3
key3="/c/temp/data.dat"
```

Les VIs de fichiers de configuration supportent les types de données suivants :

- Chaînes de caractères
- Chemins
- Booléens
- Nombres 64 bits à virgule flottante (Double)
- Entiers 32 bits signés (I32)
- Entiers 32 bits non signés (U32)

Les données de chaîne de caractères du fichier doivent être entre guillemets. Tous les caractères non imprimables de la chaîne de caractères sont stockés dans le fichier avec leurs codes hexadécimaux d'échappement équivalents (par exemple, `\0D` pour le retour chariot). De plus, les caractères de barre oblique inverse sont stockés dans le fichier comme des barres obliques inverses doubles (par exemple, `\\` pour `\`).

Les données de chemin sont stockées sous un format neutre du point de vue de la plate-forme. Ce format correspond au format standard UNIX pour les chemins. Les VIs interprètent le chemin absolu `/c/temp/données.dat` de la façon suivante pour les différentes plates-formes G :

- Windows : `c\temp\données.dat`
- MacOS : `c:temp:données.dat`
- UNIX : `/c/temp/données.dat`

De plus, les VIs interprètent le chemin relatif `temp/données.dat` de la façon suivante :

- Windows : `temp\données.dat`
- MacOS : `:temp:données.dat`
- UNIX : `temp/données.dat`

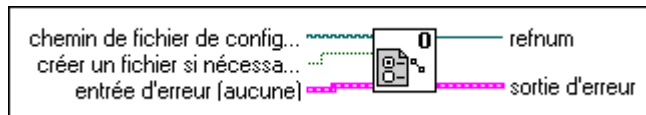
Fermer les données de Config (Close Config Data.vi)

Ferme une référence aux données de configuration identifiées par **refnum**. Si **écrire un fichier de configuration ?** est Vrai (TRUE), le VI écrit les données dans le fichier de configuration indépendant de la plate-forme identifié par **refnum**.



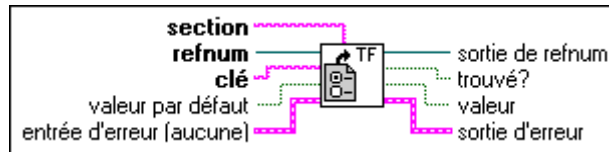
Ouvrir les données de Config (Open Config Data.vi)

Ouvre une référence aux données de configuration qui se trouvent dans un fichier de configuration indépendant de la plate-forme. Si le fichier spécifié n'existe pas et que **créer un fichier si nécessaire ?** est Vrai (TRUE), le VI crée également le fichier de configuration.



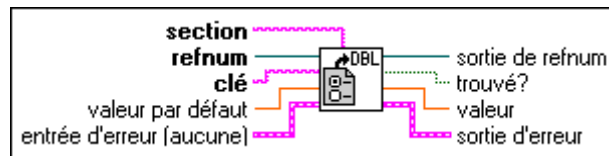
Lire une clé (booléen) (Read Key (Boolean).vi)

Lit une **valeur** booléenne associée à une **clé** dans une **section** spécifiée des données de configuration identifiées par **refnum**. Si la **clé** n'existe pas, le VI retourne la **valeur par défaut**.



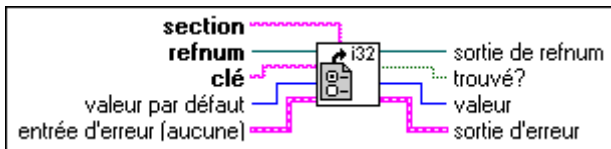
Lire une clé (double) (Read Key (double).vi)

Lit la **valeur** 64 bits à virgule flottante associée à une **clé** dans une **section** spécifiée des données de configuration identifiées par **refnum**. Si la **clé** n'existe pas, le VI retourne la **valeur par défaut**.



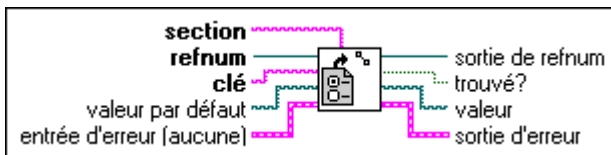
Lire une clé (I32) (Read Key (I32).vi)

Lit la **valeur** entière 32 bits signée associée à une **clé** dans une **section** spécifiée des données de configuration identifiées par **refnum**. Si la **clé** n'existe pas, le VI retourne la **valeur par défaut**.



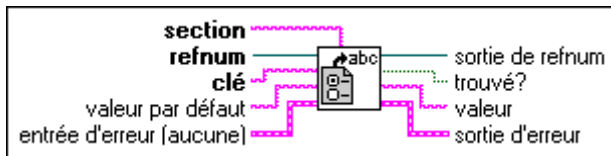
Lire une clé (chemin) (Read Key (Path).vi)

Lit la **valeur** d'un chemin associé à une **clé** dans une **section** spécifiée des données de configuration identifiées par **refnum**. Si la **clé** n'existe pas, le VI retourne la **valeur par défaut**.



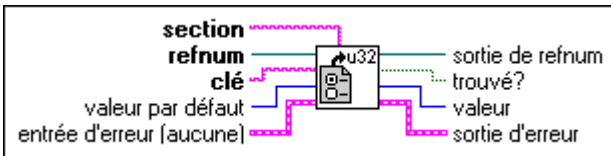
Lire une clé (chaîne de caractères) (Read Key (String).vi)

Lit la **valeur** d'une chaîne de caractères associée à une **clé** dans une **section** spécifiée des données de configuration identifiées par **refnum**. Si la **clé** n'existe pas, le VI retourne la **valeur par défaut**.



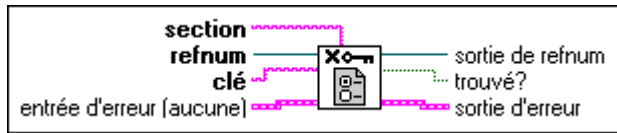
Lire une clé (U32) (Read Key (U32).vi)

Lit la **valeur** entière 32 bits non signée associée à une **clé** dans une **section** spécifiée des données de configuration identifiées par **refnum**. Si la **clé** n'existe pas, le VI retourne la **valeur par défaut**.



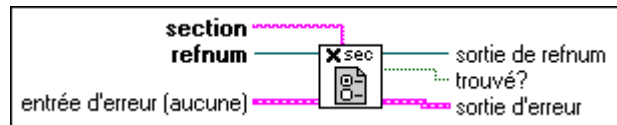
Supprimer une clé (Remove Key.vi)

Supprime une **clé** dans une **section** spécifiée des données de configuration identifiées par **refnum**.



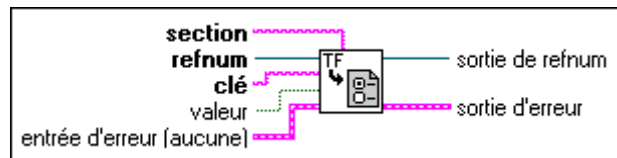
Supprimer une section (Remove Section.vi)

Supprime une **section** des données de configuration identifiées par **refnum**.



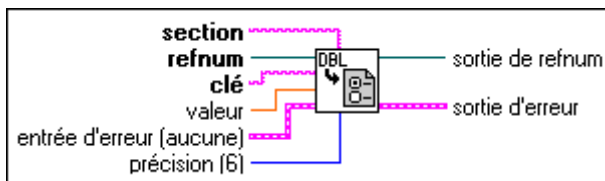
Ecrire une clé (booléen) (Write Key (Boolean).vi)

Ecrit une **valeur** booléenne associée à une **clé** dans une **section** spécifiée des données de configuration identifiées par **refnum**. Si la **clé** existe, le VI remplace la valeur existante. Si la **clé** n'existe pas, le VI ajoute la paire **clé/valeur** à la fin de la **section** spécifiée. Si la **section** n'existe pas, le VI ajoute une **section**, avec la paire **clé/valeur**, à la fin des données de configuration.



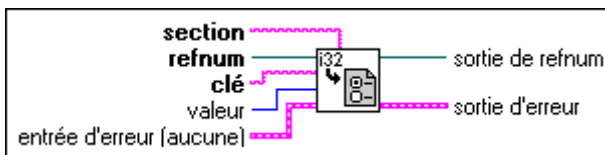
Ecrire une clé (double) (Write Key (Double).vi)

Ecrit la **valeur** 64 bits à virgule flottante associée à une **clé** dans une **section** spécifiée des données de configuration identifiées par **refnum**. Si la **clé** existe, le VI remplace la valeur existante. Si la **clé** n'existe pas, le VI ajoute la paire **clé/valeur** à la fin de la section spécifiée. Si la **section** n'existe pas, le VI ajoute une **section**, avec la paire **clé/valeur**, à la fin des données de configuration.



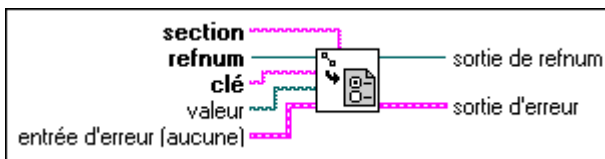
Ecrire une clé (I32) (Write Key (I32).vi)

Ecrit la **valeur** entière 32 bits signée associée à une **clé** dans une **section** spécifiée des données de configuration identifiées par **refnum**. Si la **clé** existe, le VI remplace la valeur existante. Si la **clé** n'existe pas, le VI ajoute la paire **clé/valeur** à la fin de la section spécifiée. Si la **section** n'existe pas, le VI ajoute une **section**, avec la paire **clé/valeur**, à la fin des données de configuration.



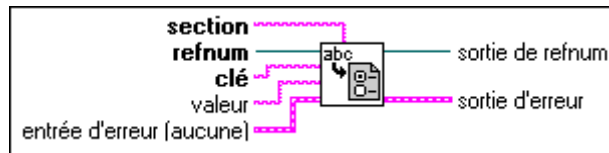
Ecrire une clé (chemin) (Write Key (Path).vi)

Ecrit la **valeur** d'un chemin associé à une **clé** dans une **section** spécifiée des données de configuration identifiées par **refnum**. Si la **clé** existe, le VI remplace la valeur existante. Si la **clé** n'existe pas, le VI ajoute la paire **clé/valeur** à la fin de la section spécifiée. Si la **section** n'existe pas, le VI ajoute une **section**, avec la paire **clé/valeur**, à la fin des données de configuration.



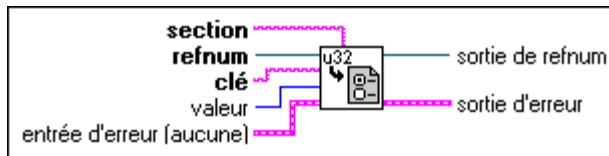
Ecrire une clé (chaîne de caractères) (Write Key (String).vi)

Ecrit la **valeur** d'une chaîne de caractères associée à une **clé** dans une section spécifiée des données de configuration identifiées par **refnum**. Si la **clé** existe, le VI remplace la valeur existante. Si la **clé** n'existe pas, le VI ajoute la paire **clé/valeur** à la fin de la section spécifiée. Si la **section** n'existe pas, le VI ajoute une **section**, avec la paire **clé/valeur**, à la fin des données de configuration.



Ecrire une clé (U32) (Write Key (U32).vi)

Ecrit la **valeur** entière 32 bits non signée associée à une **clé** dans une section spécifiée des données de configuration identifiées par **refnum**. Si la **clé** existe, le VI remplace la valeur existante. Si la **clé** n'existe pas, le VI ajoute la paire **clé/valeur** à la fin de la section spécifiée. Si la **section** n'existe pas, le VI ajoute une **section**, avec la paire **clé/valeur**, à la fin des données de configuration.



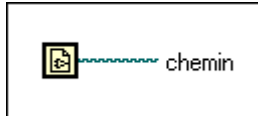
Descriptions des constantes de fichiers

Les constantes suivantes sont disponibles dans la sous-palette **Constantes de fichiers**.



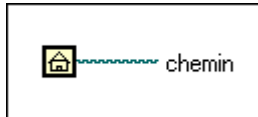
Constante de chemin du VI actuel

Retourne le chemin du fichier contenant le VI où cette fonction apparaît. Si le VI est incorporé dans une application (avec les bibliothèques de construction des applications), la fonction retourne le chemin du VI dans le fichier d'application et interprète le fichier d'application comme bibliothèque de VI.



Constante de répertoire par défaut

Retourne le chemin vers votre répertoire par défaut. Le répertoire par défaut correspond à un répertoire que la boîte de dialogue de fichier affiche au départ. La boîte de dialogue Préférences de LabVIEW (**Edition**»**Préférences**) définit ce répertoire sous **Chemins**.



Chemin vide

Retourne un chemin vide.



Pas un chemin

Retourne un chemin dont la valeur est Pas un chemin. Vous pouvez utiliser ce chemin comme sortie de structures et de sous-VIs quand une erreur survient.



Pas un refnum

Retourne un refnum dont la valeur est Pas un refnum. Vous pouvez utiliser ce refnum comme sortie de structures et de sous-VIs quand une erreur survient.



Constante de chemin

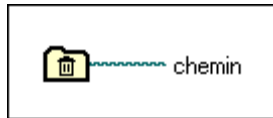
Utilisez cette constante pour fournir un répertoire ou chemin du fichier constant dans un diagramme. Définissez cette valeur en cliquant à l'intérieur de la constante avec l'outil Doigt et en tapant la valeur. Utilisez la syntaxe standard pour le chemin du fichier spécifique à votre plate-forme particulière. Vous pouvez définir la valeur de la constante de chemin sur Pas un chemin en cliquant sur le symbole de chemin avec l'outil Doigt et en sélectionnant **Pas un chemin** dans le menu qui s'affiche. Reportez-vous à la section *Chemins et refnoms*, au chapitre 6, *Chaînes de caractères et E/S sur fichiers*, dans le *Manuel de l'utilisateur de LabVIEW*, pour plus d'informations sur l'emploi de la valeur Pas un chemin.



La valeur de la constante de chemin ne peut pas être modifiée pendant l'exécution du VI. Vous pouvez affecter des étiquettes à cette constante.

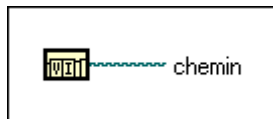
Constante de répertoire temporaire

Retourne le chemin sur votre répertoire temporaire. Le répertoire temporaire est un répertoire où vous stockez des informations passagères qui sont censées être effacées périodiquement par l'utilisateur ou le système d'exploitation. La boîte de dialogue Préférences de LabVIEW (**Edition»Préférences**), définit ce répertoire, sous **Chemins**.



Constante de bibliothèque de VI

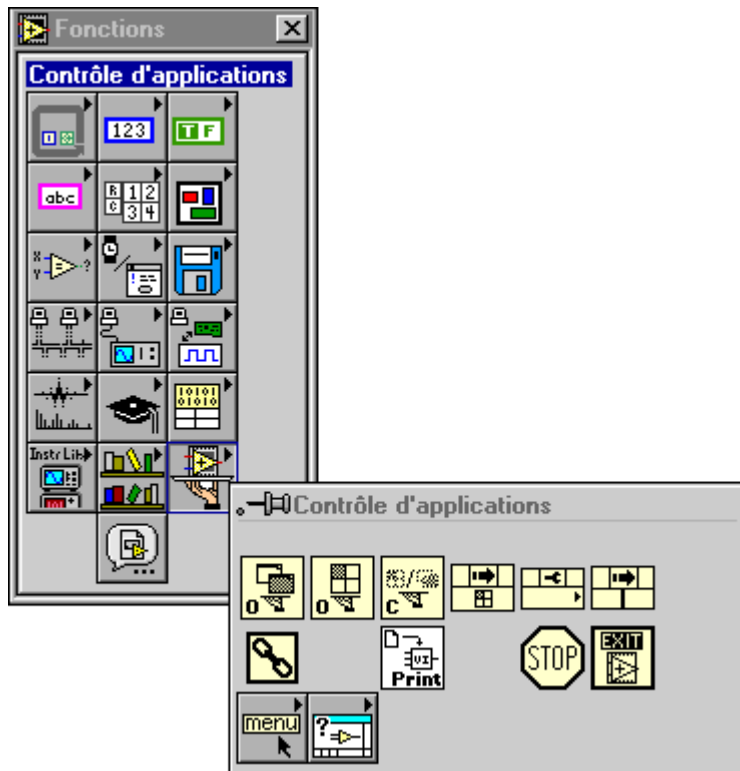
Retourne le chemin du répertoire de la bibliothèque de VI pour la bibliothèque de développement courante sur l'ordinateur utilisé. La boîte de dialogue Préférences de LabVIEW (**Edition»Préférences**), définit ce répertoire, sous **Chemins**. Si vous construisez une application en utilisant les bibliothèques de constructions d'applications, ce chemin représente le chemin du répertoire où réside l'application.



Fonctions de contrôle d'applications

Ce chapitre décrit les fonctions de contrôle d'applications.

Pour ouvrir la palette **Contrôle d'applications**, illustrée ci-après, sélectionnez **Fonctions»Contrôle d'applications**.



La palette **Contrôle d'applications** comporte les sous-palettes suivantes:

- Fonctions d'aide
- Fonctions de menus

Fonctions de contrôle d'applications

Les fonctions suivantes de contrôle d'applications sont disponibles.

Nœud d'appel par référence

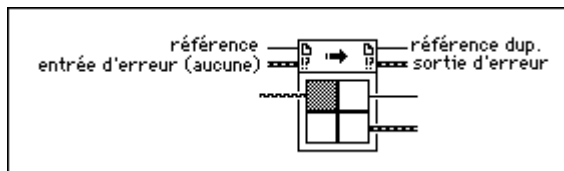
Le “Nœud d'appel par référence” est comparable au nœud d'un sous-VI : vous pouvez utiliser l'un des deux pour appeler un VI. Il existe cependant une importante différence. Avec un nœud de sous-VI, vous déterminez quel VI est appelé quand vous posez le nœud dans le diagramme.

Avec le “Nœud d'appel par référence”, l'utilisateur final détermine quel VI est appelé pendant l'exécution, par la valeur câblée à l'entrée de **référence**. Le “Nœud d'appel par référence” appelle un VI qui se trouve sur un autre ordinateur.

Quatre terminaux se trouvent dans la partie supérieure du “Nœud d'appel par référence” : une paire de terminaux de référence de VI et une paire de clusters d'erreur d'E/S transmissibles. L'entrée de référence de VI n'accepte que les connexions en provenance d'une référence de VIs de type strict. En-dessous de ces terminaux se trouve une zone où réside un cadre connecteur identique à celui du VI et qui contient ses terminaux (au lieu de son icône). Le cadre connecteur d'entrée de la référence de VI de type strict détermine le pattern et les types de données de ce cadre connecteur. Câblez ces terminaux de la même façon qu'un sous-VI normal.

Si aucun des terminaux du cadre connecteur ne comporte de fils de liaison, le cadre connecteur s'adapte automatiquement au cadre connecteur de l'entrée de la référence de VI. Néanmoins, si des fils y sont connectés, le nœud ne s'adapte pas automatiquement et vous devez changer le cadre connecteur (en brisant peut-être les liens) en ouvrant le menu local du nœud et en sélectionnant l'option **Adapter à l'entrée de référence** dans le menu.

Pendant l'exécution d'un appel de VI, il existe un petit délai supplémentaire qui n'existe pas lors d'un appel normal de sous-VI. Ce délai supplémentaire est dû à la validation de la référence de VI et à quelques autres activités. Néanmoins, lors d'un appel d'un VI dans un LabVIEW local, ce délai supplémentaire est négligeable, excepté les sous-VIs les plus petits. L'appel d'un VI situé dans une autre application LabVIEW (à travers le réseau) exige un délai supplémentaire bien plus long. L'entrée de **référence** détermine si le VI est appelé par le “Nœud d'appel par référence”.



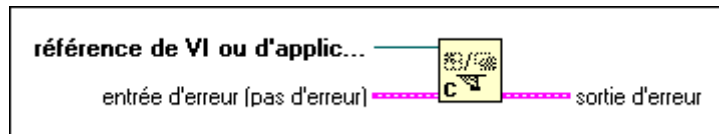
Chaîne d'appel

Retourne une référence à une application LabVIEW ou à VI.



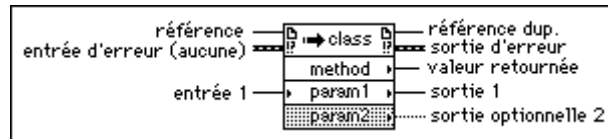
Fermer l'application ou la référence de VI

Ferme un VI ouvert ou une connexion TCP à une copie ouverte de LabVIEW.



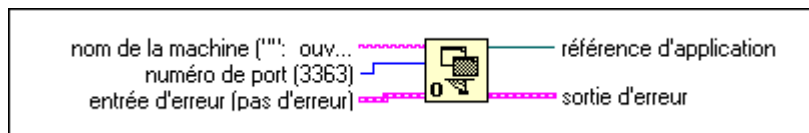
Nœud de méthode

Invoke une méthode ou action relative à un VI. La plupart des méthodes comportent des paramètres qui y sont associés. Pour sélectionner la méthode, ouvrez le menu local n'importe où sur le nœud et sélectionnez **Méthodes**. Une fois la méthode sélectionnée, les paramètres associés apparaissent comme dans l'illustration suivante. Vous pouvez définir et obtenir les valeurs des paramètres. Les paramètres dont l'arrière-plan est blanc représentent des entrées requises et les paramètres dont l'arrière-plan est gris représentent des entrées recommandées.



Ouvrir une référence d'application

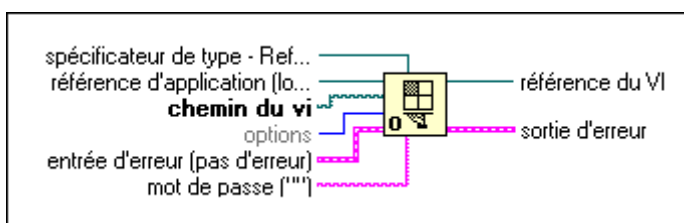
Retourne une référence à une application VI serveur exécutée sur un ordinateur précis. Si vous ne spécifiez pas une valeur pour le **nom de la machine**, cette fonction retourne alors une référence à l'application locale de LabVIEW dans laquelle cette fonction s'exécute.



Utilisez la sortie **référence d'application** comme entrée des nœuds Propriété et Invoker pour obtenir ou définir des propriétés et pour invoquer des méthodes relatives à l'application. Si vous l'utilisez comme entrée à la fonction "Ouvrir une référence de VI", vous pourrez obtenir des références à des VIs contenus dans cette application. Fermez la référence avec la fonction "Fermer l'application ou la référence de VI". Si vous oubliez de fermer cette référence, elle se ferme automatiquement quand le VI de niveau principal associé à cette fonction finit son exécution. Néanmoins, nous vous conseillons de conserver les ressources maintenant la connexion en fermant la référence une fois que vous ne l'utilisez plus.

Ouvrir une référence de VI

Retourne la référence d'un VI spécifié par la chaîne de caractères du nom ou du chemin de l'emplacement du VI sur le disque.



Vous pouvez obtenir des références à des VIs d'une autre application LabVIEW en câblant une **référence d'application** (obtenue par le biais de la fonction "Ouvrir une référence d'application") à cette fonction. Dans ce cas, l'entrée **chemin du VI** se rapporte au système de fichiers de l'ordinateur de l'application de LabVIEW à distance. Si vous câblez une référence à l'application locale de LabVIEW, vous obtenez le même comportement que si vous n'aviez rien câblé à l'entrée de la **référence d'application**.

Si vous souhaitez effectuer des opérations d'édition sur le VI référencé et que le VI comprend un diagramme protégé par un mot de passe, vous pouvez taper le mot de passe dans la chaîne de caractères d'entrée **mot de passe**. Si vous ne tapez pas le bon mot de passe, la fonction "Ouvrir une référence de VI" retourne une erreur et une référence de VI non valide. Si vous ne tapez pas de mot de passe quand vous ouvrez la référence d'un VI protégé par un mot de passe, vous pouvez obtenir la référence mais ne pouvez pas éditer le VI.

Si vous souhaitez appeler le VI spécifié par le "Nœud appel par référence", câblez une référence de VI de type strict à l'entrée **spécificateur de type**. La fonction ignore la valeur de cette entrée. Seul le type d'entrée (informations sur le cadre connecteur) est utilisé. En spécifiant ce type, la fonction "Ouvrir une référence de VI" vérifie pendant l'exécution que le cadre connecteur du VI référencé correspond à l'entrée du **spécificateur de type**.



Remarque *Il est possible de câbler un type de refnum de VI Générique à l'entrée du spécificateur de type. Ceci conduit au même comportement que si vous n'aviez pas câblé l'entrée du spécificateur de type.*

Si vous câblez l'entrée du spécificateur de type avec un refnum de VI de type strict, le VI doit satisfaire plusieurs exigences pour que la référence de VI soit correctement retournée :

- Le VI ne doit pas être brisé.
- Le VI doit être exécutable en tant que sous-VI ; autrement dit, il ne doit pas agir comme un VI de niveau principal (sauf si le VI est ré-entrant).
- Le cadre connecteur du VI doit correspondre au spécificateur de type.

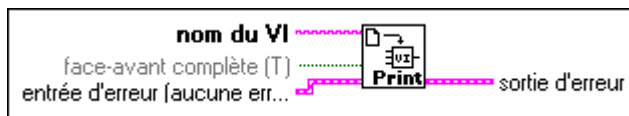
Si vous oubliez de fermer cette référence, elle se ferme automatiquement quand le VI de niveau principal associé à cette fonction finit son exécution. Néanmoins, nous vous conseillons de conserver les ressources maintenant la connexion en fermant la référence une fois que vous ne l'utilisez plus.

Si vous obtenez une référence de type strict à un VI qui est ré-entrant, un espace de données dédié est attribué à cette référence. Cet espace de données est toujours utilisé en conjonction avec la référence de VI de sortie. Ceci peut provoquer des comportements inhabituels pour les utilisateurs de LabVIEW. Par exemple, les appels parallèles (par le biais du "Nœud d'appel par référence") à un VI qui est ré-entrant en utilisant la même référence de VI ne s'exécutent pas en parallèle mais en série, l'un après l'autre.

Notez qu'une référence de VI est comparable à la fonction Pointeur des autres langages. Néanmoins, dans LabVIEW, ces pointeurs de fonctions servent également à appeler des VIs à travers le réseau.

Imprimer la face-avant (Print Panel.vi)

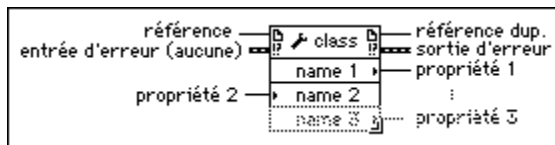
Produit la même impression que l'impression finale par programmation mais peut être appelée depuis d'autres VIs et à des moments autres que la fin de l'exécution. Cette fonction imprime, par défaut, la face-avant complète et pas seulement ce qui est visible dans la fenêtre. Ce VI suppose que le VI est chargé et n'exige pas que la fenêtre soit ouverte.



Nœud de propriété

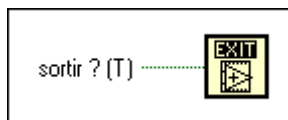
Définit (écrit) ou obtient (lit) les informations de propriété relatives à un VI et à une application. Pour sélectionner une classe de VI ou d'application, ouvrez le menu local du nœud et faites votre choix à partir du sous-menu **Sélectionner une classe de VI Serveur**. Pour sélectionner une classe d'application, sélectionnez **Application**. Pour sélectionner une classe de VI, sélectionnez **Instrument virtuel** ou câblez le refnum de VI ou d'application à **référence** et les options du nœud changent comme il se doit.

Pour sélectionner une propriété spécifique, ouvrez le menu local sur un des terminaux de **nom** et sélectionnez **Propriétés**. Pour définir des informations de propriété, ouvrez le menu local et sélectionnez **Changer en écriture** et pour obtenir des informations de propriété, ouvrez le menu local et sélectionnez **Changer en lecture**. Quelques propriétés sont en lecture seule et vous ne verrez pas apparaître **Changer en écriture** dans le menu local. Le nœud de propriété fonctionne de la même façon que les Attribute nodes. Si vous souhaitez ajouter des articles au nœud, ouvrez le menu local et sélectionnez **Ajouter un élément** ou cliquez et faites glisser le nœud pour étendre son nombre d'éléments. Quand le nœud s'exécute, les propriétés sont traitées dans l'ordre, de haut en bas. Si une erreur survient dans une des propriétés, le nœud s'arrête sur cette propriété et retourne une erreur. Les propriétés suivantes ne seront pas traitées. La chaîne d'erreur indique la propriété responsable de l'erreur. Souvenez-vous que si la petite flèche de direction sur une propriété est à gauche, vous définissez la valeur de la propriété. Si la petite flèche de direction sur une propriété est à droite, vous lisez la valeur de la propriété. Chaque nom de propriété comporte un nom court ou long qui peut être modifié en ouvrant le menu local et en sélectionnant **Format du nom**. Dans le format "Aucun nom", seul le type est affiché pour chaque propriété.



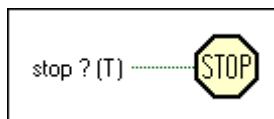
Quitter LabVIEW

Arrête tous les VIs en exécution et termine la session en cours de LabVIEW. Cette fonction ne ferme que LabVIEW ; elle n'affecte pas les autres applications. La fonction arrête tous les VIs en cours d'exécution de la même manière que la fonction Arrêter.



Arrêter

Arrête le VI dans lequel la fonction s'exécute, tout comme si vous cliquez sur le bouton Stop dans la barre d'outils. Si vous câblez l'entrée, l'arrêt n'a lieu que si sa valeur est égale à TRUE. Si vous ne câblez pas l'entrée, l'arrêt a lieu dès que le nœud courant termine son exécution.



Si vous souhaitez interrompre l'exécution de tous les VIs d'une hiérarchie à partir du diagramme, vous pouvez utiliser cette fonction mais faites-le avec prudence. Avant d'appeler la fonction Arrêter avec une entrée Vrai (TRUE), assurez-vous d'avoir d'abord fini toutes les tâches du VI, telles que fermer les fichiers, définir les valeurs d'enregistrement pour les périphériques contrôlés et ainsi de suite. Si vous placez la fonction Arrêter dans un sous-VI, vous devez clairement spécifier son comportement aux autres utilisateurs du VI car cette fonction peut faire que leurs hiérarchies de VIs interrompent leur exécution.

En général, évitez d'utiliser la fonction Arrêter quand vous avez un protocole de terminaison intégré dans votre VI. Par exemple, les opérations d'E/S doivent s'effectuer dans des boucles While pour que le VI puisse terminer la boucle sur une E/S d'erreur. Envisagez également d'utiliser une commande pour terminer la boucle à la demande de l'utilisateur au lieu de la fonction Arrêter.

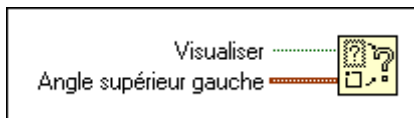
Descriptions des fonctions d'aide

L'illustration suivante montre les options disponibles dans la sous-palette **Aide**.



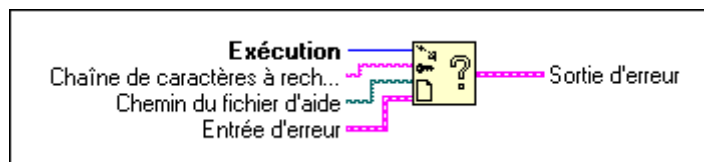
Contrôle de la fenêtre d'aide

Modifie la fenêtre d'aide en la montrant, en la masquant ou en la repositionnant.



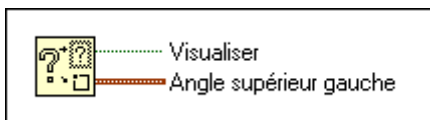
Contrôle de l'aide en ligne

Contrôle le système d'aide en ligne en affichant la table des matières du fichier d'aide, en ouvrant une section précise du fichier d'aide ou en fermant le système d'aide en ligne.



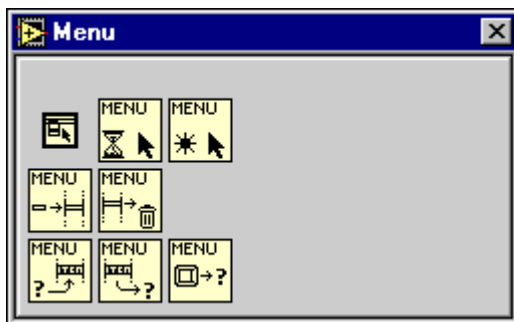
Etat de la fenêtre d'aide

Retourne des informations sur l'état et la position de la fenêtre d'aide.



Fonctions de menus

L'illustration suivante présente les options disponibles dans la sous-palette **Menu**.

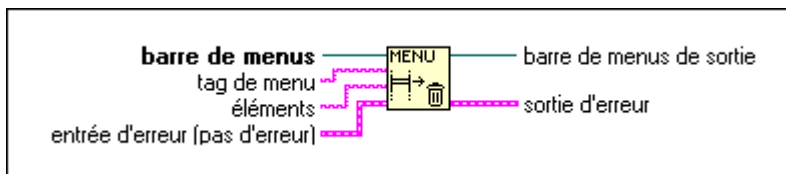


Les fonctions de menus agissent sur des menus identifiés par un refnum. Le refnum du menu d'un VI est obtenu par le biais de la constante "Barre de menus du VI actuel". Les éléments de menu sont identifiés par un tag d'élément (chaîne de caractères) ou par un chemin d'élément (chaîne de caractères), correspondant à une liste de tags d'éléments représentant le chemin depuis la racine du menu jusqu'à l'élément, et séparés par deux-points (:).

Les fonctions de menus suivantes sont disponibles.

Supprimer des éléments de menus

Supprime des éléments de menus dans la barre de menus ou dans un de ses sous-menus.

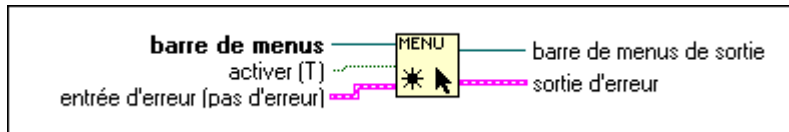


Si un **tag de menu** est spécifié, les éléments sont effacés dans le sous-menu spécifié par **tag de menu**, sinon les éléments sont effacés dans la barre de menus. La fonction retourne une erreur si le **tag de menu** ou un des éléments spécifiés n'est pas trouvé.

Les **éléments** peuvent être le tag (chaîne de caractères) d'un élément existant, un tableau de tags d'éléments existants, un indice de position (entier de base zéro) d'un élément dans le menu ou un tableau d'indices de position d'éléments dans le menu. Si vous ne câblez pas l'entrée **éléments**, tous les éléments du menu sont effacés. S'il existe un sous-menu dans n'importe lequel des éléments spécifiés, le sous-menu et tout son contenu sont effacés automatiquement. Etant donné que les séparateurs ne comportent pas de tags uniques, il vaut mieux les effacer en utilisant leurs indices de position.

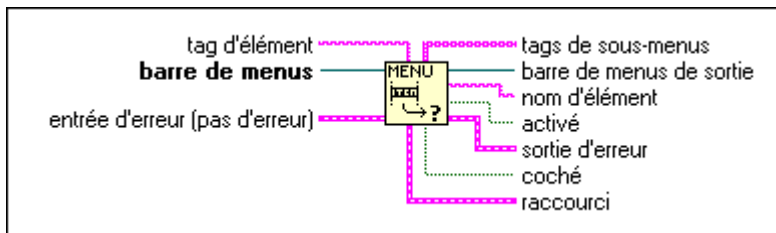
Activer le repérage de menus

Active ou désactive le repérage des sélections de menus.



Obtenir des infos sur un élément de menu

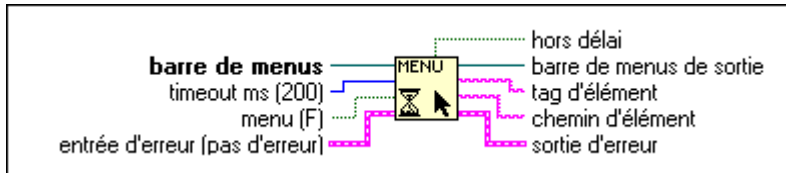
Retourne les attributs de l'élément de menu spécifié par le **tag d'élément**.



Les attributs d'éléments sont le **nom de l'élément** (la chaîne de caractères qui apparaît dans le menu), **activé** (Faux (FALSE) indique que l'élément est grisé), **coché** (spécifie s'il y a une marque à côté de l'élément) et **raccourci** (accélérateur de touche). Si l'élément est un sous-menu, ses tags d'éléments sont retournés comme un tableau de chaînes de caractères dans **tag de sous-menus**. Si **tag d'élément** n'est pas câblé, la fonction retourne les éléments de la barre de menus. Si **tag d'élément** n'est pas valide, la fonction retourne une erreur.

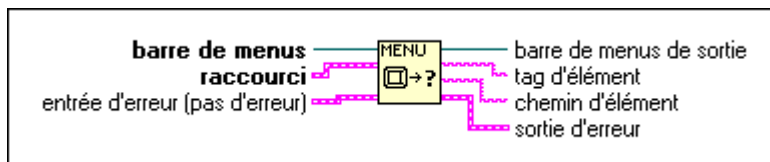
Élément de menu sélectionné

Retourne le **tag d'élément** du dernier élément de menu sélectionné ou attend pendant une durée égale à `timeout` en ms. Le **chemin d'élément** est une chaîne de caractères décrivant la position de l'élément dans la hiérarchie du menu, dont le format est une liste de tags de menus séparés par des deux-points (:). Si **menu** est paramétré sur Vrai (True), la sélection de menus est bloquée après la lecture d'un tag d'élément.



Infos sur un raccourci de menus

Retourne l'élément de menu accessible par le biais d'un raccourci donné.

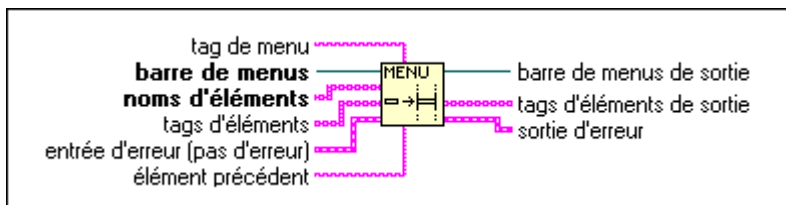


Le **chemin d'élément** est une chaîne de caractères de tags d'éléments de menus séparés par deux-points (:).

Le **raccourci** est une chaîne de caractères (touche) et un booléen (spécifie si le raccourci comprend la touche Maj).

Insérer des éléments de menu

Insère des éléments de menu dans une barre de menus ou dans le sous-menu d'une barre de menus.



Le **tag de menu** spécifie le sous-menu où les éléments doivent être insérés. Si vous ne spécifiez pas le **tag de menu**, les éléments sont insérés dans la **barre de menus**.

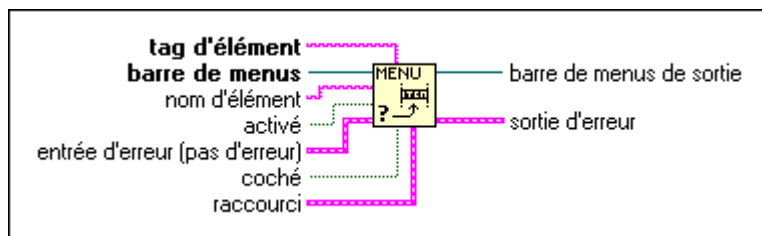
Les **noms d'éléments** et les **tags d'éléments** identifient les éléments à insérer dans le menu. Le type de **noms d'éléments** et de **tags d'éléments** peut être un tableau de chaînes de caractères (pour insérer plusieurs éléments) ou juste une chaîne de caractères (pour insérer un seul élément). Vous pouvez câbler des **noms d'éléments** ou des **tags d'éléments**. Dans ce cas, les noms et les tags ont les mêmes valeurs. Si vous préférez que chaque élément ait un nom et un tag différents, vous devez câbler des valeurs séparées pour les **noms d'éléments** et les **tags d'éléments**.

Élément précédent spécifie la position où les éléments sont insérés. **Élément précédent** peut être le tag (chaîne de caractères) d'un élément existant ou un indice de position (entier de base zéro) dans le menu. Pour insérer un élément au début du menu, câblez un nombre plus petit que 0 au paramètre **Élément précédent**. Pour insérer un élément à la fin du menu, câblez un nombre supérieur au nombre d'éléments dans le menu. Vous pouvez insérer un séparateur avec le tag d'application APP_SEPARATOR. La fonction vérifie toujours que les tags de tous les éléments de menu insérés sont uniques dans la hiérarchie du menu en ajoutant des chiffres aux tags fournis, si nécessaire.

Le paramètre **tags d'éléments de sortie** retourne les tags courants des éléments insérés. Si **tag de menu** ou **Élément précédent** (tag) n'est pas trouvé, la fonction retourne une erreur.

Définir les infos de l'élément de menu

Définit les attributs d'un élément de menu spécifié par le biais de **menu** et de **tag d'élément**. Les attributs d'éléments sont **nom de l'élément** (la chaîne de caractères qui apparaît dans le menu), **activé** (Faux (FALSE) indique que l'élément est grisé), **coché** (spécifie s'il y a une marque à côté de l'élément) et **raccourci** (accélérateur de touche). Les attributs qui ne sont pas câblés ne sont pas modifiés. Si **tag d'élément** n'est pas valide, la fonction retourne une erreur.

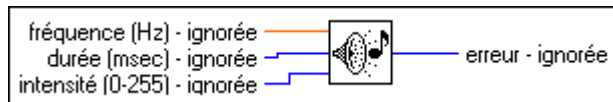


Descriptions des fonctions avancées

Les fonctions avancées suivantes sont disponibles.

Bip (Beep.vi)

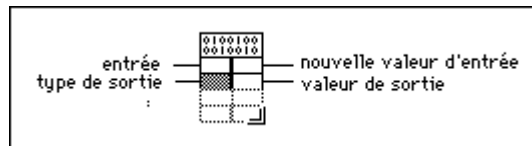
Demande au système de produire une tonalité sonore. Vous pouvez spécifier la fréquence de la tonalité en Hertz, la durée en millisecondes et l'intensité comme valeur comprise entre 0 et 255, où 255 est la tonalité la plus forte. Bien que ce VI apparaisse sur toutes les plates-formes, les paramètres de fréquence, de durée et d'intensité ne fonctionnent que sur un Macintosh.



Code Interface Node

Appelle un code écrit dans un langage de programmation conventionnel, comme le C, directement à partir d'un diagramme. Les Code Interface Node (CIN) vous permettent d'utiliser des algorithmes écrits dans un autre langage ou d'accéder aux fonctions spécifiques d'une plate-forme ou d'un équipement que le G ne supporte pas directement.

Les CIN sont redimensionnables et affichent les types de données de leurs entrées et de leurs sorties connectées, de façon similaire à la fonction Assembler. L'illustration suivante montre la fonction CIN.



L'interface de LabVIEW vers des codes externes est très puissante. Vous pouvez transmettre n'importe quel nombre de paramètres depuis ou vers un code externe et chaque paramètre peut être de n'importe quel type de données G. LabVIEW fournit plusieurs bibliothèques de routines qui simplifient la manipulation des types de données G. Ces routines prennent en charge l'allocation de mémoire, la manipulation de fichiers et la conversion de types de données.

Si vous convertissez un VI contenant un CIN sur une autre plate-forme, vous devez recompiler le code pour la nouvelle plate-forme car les CIN utilisent un code compilé dans un autre langage de programmation. Vous pouvez écrire par ailleurs un code source pour un CIN de façon à ce qu'il ne soit pas dépendant de la machine, et n'exige qu'une recompilation pour le convertir sur une autre plate-forme. Pour obtenir des exemples de CIN, reportez-vous à `examples\cins`.

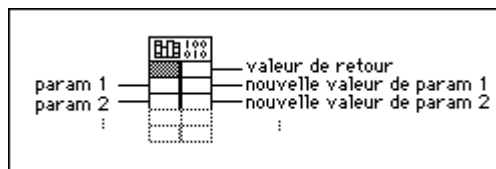
Pour plus d'informations sur Code Interface Node, consultez *Code Interface Reference Manual de LabVIEW*, disponible uniquement en format de document portable PDF (Portable Document Format).

Fonction “Appeler une fonction d'une DLL”

Appelle des bibliothèques standard sans écrire un Code Interface Node (CIN). Sous Windows, vous pouvez appeler directement une fonction de bibliothèque à lien dynamique (DLL). Sous Macintosh et UNIX, vous pouvez appeler directement une fonction de bibliothèque partagée. Sous Macintosh 68K, vous devez installer l'extension de système CFM-68K pour que le nœud de la fonction “Appeler une fonction d'une DLL” puisse fonctionner.

Ce nœud supporte un grand nombre de types de données et de conventions d'appel. Vous pouvez l'utiliser pour appeler des fonctions dans la plupart des bibliothèques standard et personnalisées.

Le nœud de la fonction “Appeler une fonction d'une DLL”, illustré ci-dessous, est comparable au nœud de la fonction Code Interface Node.

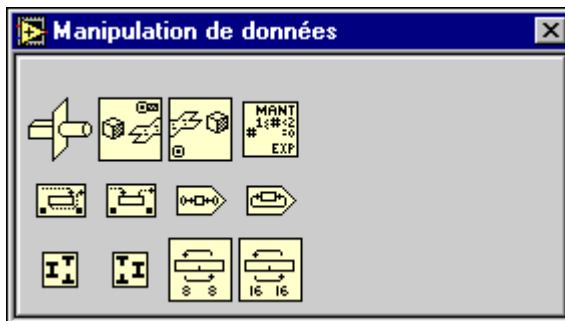


La fonction “Appeler une fonction d'une DLL” comporte une paire de terminaux d'entrée/sortie avec les entrées à gauche et les sorties à droite. Vous pouvez utiliser une paire ou les deux. La valeur de sortie pour la fonction est retournée dans le terminal droit de la paire située en haut du nœud. S'il n'y a pas de valeur retournée, la paire de terminaux n'est pas utilisée. Chaque paire de terminaux supplémentaire correspond à un paramètre de la liste de paramètres de la fonction. Vous pouvez transmettre une valeur à la fonction en câblant le terminal gauche de la paire. Vous pouvez lire la valeur d'un paramètre après l'appel de la fonction en câblant le terminal droit de la paire.

Si vous sélectionnez **Configurer...** dans le menu local du nœud, vous voyez apparaître une boîte de dialogue où vous pouvez spécifier le nom de la bibliothèque ou du chemin, le nom de la fonction, les conventions d'appel, les paramètres et la valeur de sortie du nœud. Quand vous cliquez sur **OK**, le nœud s'accroît automatiquement pour atteindre le nombre correct de terminaux. Il configure ensuite les terminaux sur les types de données corrects. Pour plus d'informations sur la fonction “Appeler une fonction d'une DLL”, reportez-vous au chapitre 25, *Codes d'appel d'autres langages*, dans le *Manuel de référence de programmation en G*.

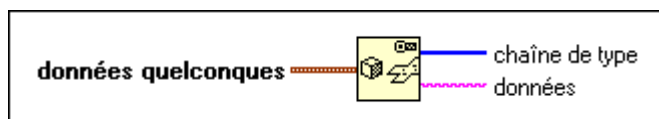
Descriptions des fonctions de manipulation de données

L'illustration ci-après montre les options disponibles dans la sous-palette **Manipulation de données**.



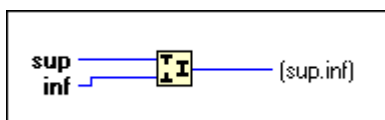
Aplatir en chaîne de caractères

Convertit des **données quelconques** en chaîne de caractères de valeurs binaires. **Type de chaîne de caractères** est un descripteur de type qui décrit le type des **données quelconques**. **Chaîne de caractères de données** constitue la forme aplatie des **données quelconques**. Pour plus d'informations sur les descripteurs de types et les données aplaties, reportez-vous à la section *Données aplaties*, dans l'Annexe A, *Formats de stockage de données*, du *Manuel de référence de programmation en G*.



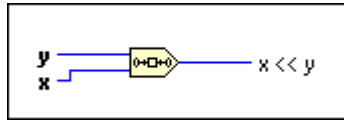
Regrouper des nombres

Crée un nombre à partir de composantes de type octet ou mot.



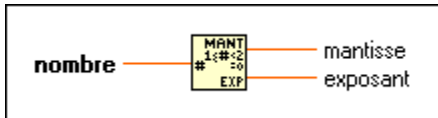
Décalage logique

Effectue un décalage de **x**, sur le nombre de bits spécifié par **y**.



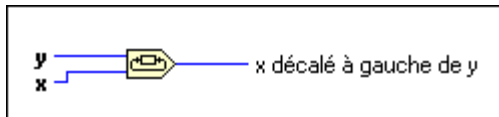
Mantisse & Exposant

Retourne la mantisse et l'exposant de la valeur numérique d'entrée de façon à ce que **nombre = mantisse * 2^{exposant}**. Si **nombre** est 0, **mantisse** et **exposant** sont égaux à 0. Sinon, la valeur de **mantisse** est supérieure ou égale à 1 et inférieure à 2, et la valeur de l'**exposant** est un entier.



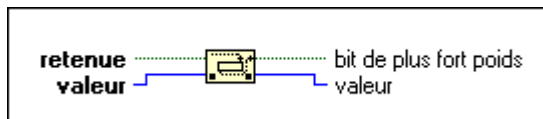
Rotation

Effectue une rotation de **x**, sur le nombre de bits spécifié par **y**.



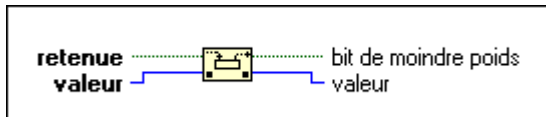
Rotation gauche avec retenue

Effectue une rotation de chaque bit de la **valeur** d'entrée vers la gauche (du bit de moindre poids au bit du plus fort poids), insère une **retenue** dans le bit d'ordre inférieur et retourne le bit du plus fort poids.



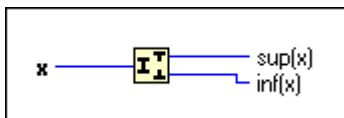
Rotation droite avec retenue

Effectue la rotation de chaque bit de la valeur d'entrée vers la droite (du bit de moindre poids au bit du plus fort poids), insère une **retenue** dans le bit d'ordre supérieur et retourne le bit de moindre poids.

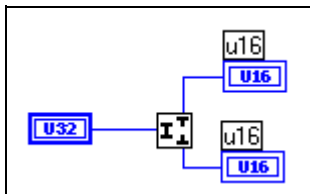
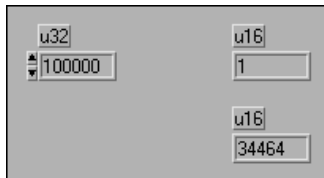


Eclater un nombre

Divise un nombre en composantes de type octet ou mot.

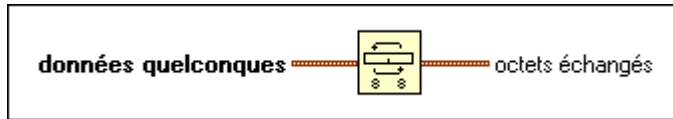


L'illustration suivante montre un exemple de l'utilisation de la fonction "Eclater un nombre". La fonction éclate le nombre de 32 bits signé 100 000 en une composante de mot supérieur 1 et une composante de mot inférieur 34 464.



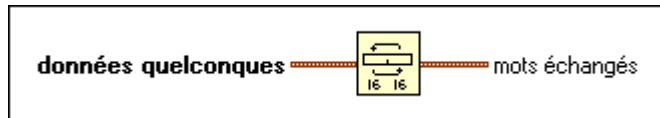
Intervertir les octets

Intervertit les 8 bits d'ordre supérieur et les 8 bits d'ordre inférieur pour chaque mot de données quelconques.



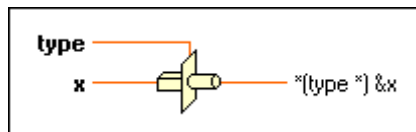
Intervertir les mots

Intervertit les 16 bits d'ordre supérieur et les 16 bits d'ordre inférieur pour chaque mot de données quelconques.



Adapter le type

Adapte **x** aux types de données, **type**.



Adapter des données à une chaîne de caractères les convertit au format “big endian” indépendant de la machine. Autrement dit, la fonction place l’octet ou le mot de plus fort poids en premier et l’octet ou le mot de moindre poids en dernier. Elle supprime l’alignement et convertit les nombres à précision étendue en 16 octets. Adapter une chaîne de caractères à un tableau 1D convertit la chaîne de caractères du format indépendant de la machine au format natif de la plate-forme.

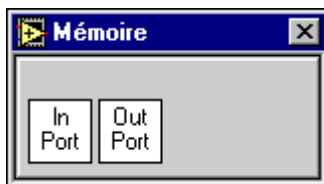
Redresser à partir d'une chaîne de caractères

Convertit une **chaîne binaire** au type câblé à l'entrée **type**. Cette fonction exécute l'inverse de la fonction "Aplatir en chaîne de caractères". La **chaîne binaire** doit contenir des données aplaties du type câblé à l'entrée **type**. Pour plus d'informations sur les descripteurs de types et les données aplaties, reportez-vous à la section *Données aplaties*, dans l'Annexe A, *Formats de stockage de données*, du *Manuel de référence de programmation en G*.



Descriptions des VIs de mémoire

L'illustration suivante montre les options disponibles dans la sous-palette **Mémoire**.



Port d'entrée (In Port.vi) (Windows 3.1 et Windows 95)

Lit un entier de type octet ou mot à une **adresse de registre** spécifique. Etant donné que ce VI n'est pas disponible sur toutes les plates-formes, les VIs utilisant ce sous-VI ne sont pas portables.



Port de sortie (Out Port.vi) (Windows 3.1 et Windows 95)

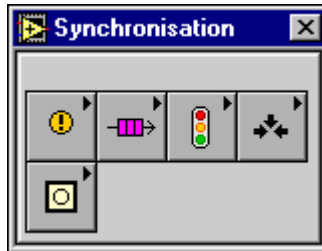
Ecrit un entier de type octet ou mot à une **adresse de registre** spécifique. Etant donné que ce VI n'est pas disponible sur toutes les plates-formes, les VIs utilisant ce sous-VI ne sont pas portables.



VIs de synchronisation

Vous pouvez synchroniser des tâches s'exécutant en parallèle avec les VIs de synchronisation. Vous pouvez également utiliser les VIs de synchronisation pour transmettre des données entre tâches parallèles. Vous pouvez ouvrir la palette **Synchronisation** en sélectionnant **Fonctions»Avancé»Synchronisation**.

L'illustration suivante montre les options disponibles dans la palette **Synchronisation**.



La palette **Synchronisation** comporte cinq sous-palettes :

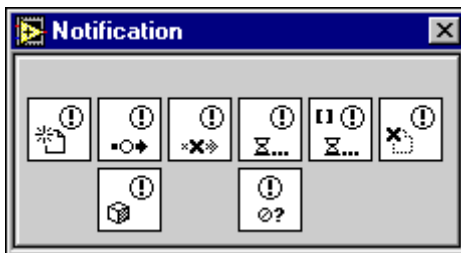
- VIs Notification
- VIs File d'attente
- VIs Rendez-vous
- VIs Sémaphore
- Fonctions d'occurrence

VIs Notification

Vous pouvez utiliser les VIs Notification pour transmettre des données d'une tâche à une ou plusieurs autres tâches parallèles. En particulier, vous pouvez utiliser ces VIs quand vous souhaitez que des VIs ou des parties du diagramme attendent qu'un autre VI ou qu'une autre partie du diagramme transmette des données.

Les VIs Notification sont différents des VIs File d'attente en ce que les données transmises ne sont pas bufférisées. Autrement dit, si aucun élément n'attend de notification lorsque celle-ci est envoyée, les données sont "perdus" si une autre notification est envoyée. Notez également que plusieurs VIs "Attendre une notification" (Wait on Notification.vi) peuvent recevoir les mêmes données.

Vous pouvez accéder aux VIs Notification en sélectionnant **Fonctions»Avancé»Synchronisation»Notification**.



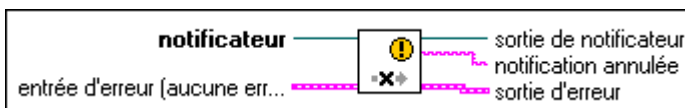
Les VIs Notification utilisent la commande **refnum de notificateur** située dans la palette **Commandes»Chemin & Refnum**.



Le refnum de notificateur peut être utilisé avec les VIs suivants.

Annuler une notification (Cancel Notifier.vi)

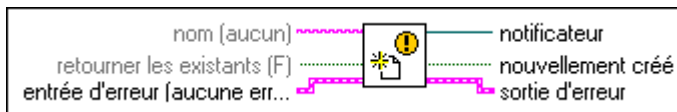
Annule et retourne une notification envoyée antérieurement.



Ce VI évite d'appeler le VI "Attendre une notification" (Wait on Notification.vi) avec le paramètre **ignorer précédent** réglé sur Faux (FALSE) pour déterminer la notification envoyée précédemment.

Créer un notificateur (Create Notifier.vi)

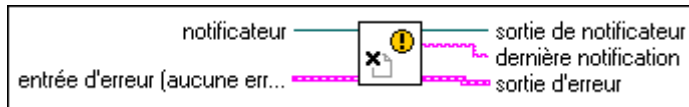
Recherche un **notificateur** existant ou crée un nouveau **notificateur** et retourne un refnum que vous pouvez utiliser quand vous appelez d'autres VIs Notification.



Si le **nom** est spécifié, le VI cherche d’abord un **notificateur** existant portant le même **nom** et retourne son refnum s’il existe. Si aucun notificateur portant le même nom n’existe et que l’entrée **retourner les existants** est Faux (FALSE), le VI crée un nouveau **notificateur** et retourne son refnum. La sortie **nouvellement créé** retourne Vrai (TRUE) si le VI crée un nouveau notificateur.

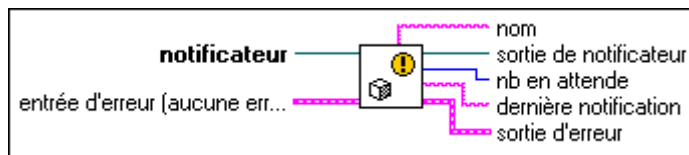
Détruire un notificateur (Destroy Notifier.vi)

Détruit le **notificateur** spécifié et retourne la **dernière notification** transmise. Tous les VIs “Attendre une notification” (Wait on Notification.vi) qui attendent ce notificateur sont immédiatement suspendus et retournent une erreur.



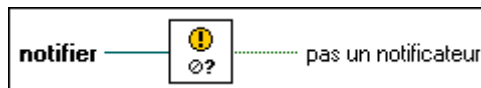
Obtenir l’état d’un notificateur (Get Notifier Status.vi)

Retourne les informations sur l’état courant du **notificateur**.



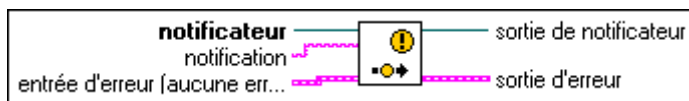
Pas un notificateur (Not a Notifier.vi)

Retourne Vrai (TRUE) si le **notificateur** n’est pas un refnum de notificateur valide.



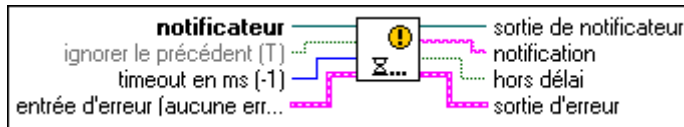
Envoyer une notification (Send Notification.vi)

Envoie une **notification** au **notificateur** spécifié. Tous les VIs “Attendre une notification” (Wait on Notification.vi) qui attendent ce **notificateur** arrêtent d’attendre et retournent la **notification** spécifiée.



Attendre une notification (Wait on Notification.vi)

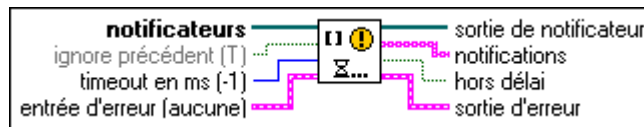
Attend que le VI “Envoyer une notification” (send Notification.vi) envoie une **notification** au notificateur spécifié.



Si **ignorer précédent** est Faux (FALSE) et qu’une notification a été envoyée depuis le dernier appel de ce VI, le VI revient immédiatement à la valeur de l’ancienne notification, avec **hors délai** sur Faux (FALSE). Si l’entrée **ignorer précédent** est Vrai (TRUE), le VI attend pendant une durée égale à timeout en millisecondes (–1 par défaut ou indéfini) avant de suspendre l’exécution. Si une notification est envoyée, **hors délai** retourne Faux (FALSE). Si aucune notification n’est envoyée ou si le **notificateur** n’est pas valide, **hors délai** retourne Vrai (TRUE).

Attendre une notification multiple (Wait on Notification from Multiple.vi)

Attend que le VI “Envoyer une notification” (Send Notification.vi) envoie une notification à l’un des notificateurs spécifiés.



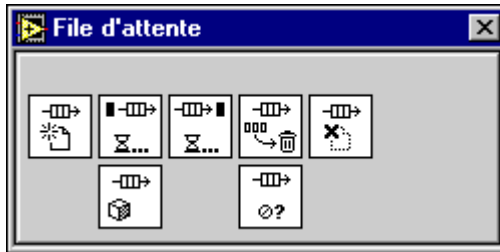
Si **ignorer précédent** est Faux (FALSE) et qu’une notification a été envoyée à l’un des notificateurs depuis le dernier appel de ce VI, le VI revient immédiatement à la valeur ou aux valeurs de la ou des anciennes notifications avec **hors délai** sur Faux (FALSE). Si l’entrée **ignorer précédent** est Vrai (TRUE), le VI attend pendant une durée égale à timeout en millisecondes (–1 par défaut ou indéfini) avant de suspendre l’exécution. Si au moins une notification est envoyée, **hors délai** retourne Faux (FALSE). Si aucune notification n’est envoyée, **hors délai** retourne Vrai (TRUE).

VIs File d’attente

Vous pouvez utiliser les VIs File d’attente pour transmettre une séquence ordonnée d’éléments de données d’une tâche à une autre tâche parallèle. Vous pouvez en particulier utiliser ces VIs quand vous souhaitez qu’une tâche attende qu’une autre tâche lui fournisse des données. Vous pouvez également utiliser ces VIs quand vous souhaitez qu’une tâche attende qu’une autre tâche ait traité des données fournies par la première tâche.

Les VIs File d'attente sont différents des VIs Notification en ce que les données sont transmises par l'intermédiaire d'un buffer. Autrement dit, si un élément inséré dans la file d'attente n'est pas lu, celui-ci reste dans la file d'attente jusqu'à ce qu'il soit supprimé. De plus, si des données sont insérées dans une file d'attente et si deux VIs parallèles attendent de les supprimer, un seul VI reçoit des données.

Accédez aux VIs File d'attente en sélectionnant **Fonctions»Avancé»Synchronisation»File d'attente**.



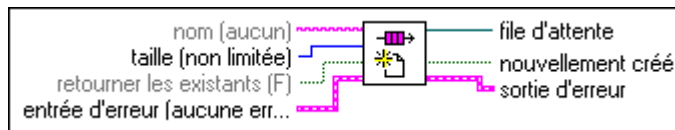
Les VIs File d'attente utilisent la commande du **refnum de file d'attente** située dans la palette **Commandes»Chemin & Refnum**.



Un refnum de file d'attente peut être utilisé avec les VIs suivants.

Créer une file d'attente (Create Queue.vi)

Cherche une file d'attente existante ou crée une nouvelle file d'attente et retourne un refnum que vous pouvez utiliser quand vous appelez d'autres VIs File d'attente.

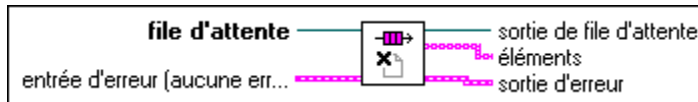


Si vous spécifiez une taille > 0, la file d'attente est limitée à cette taille. Si le VI "Insérer un élément de file d'attente" (Insert Queue Element.vi) essaie d'insérer un élément dans une file d'attente comblée, il doit attendre jusqu'à ce qu'un élément soit supprimé avec le VI "Supprimer un élément d'une file d'attente" (Remove Queue Element.vi). La taille par défaut est -1, ce qui définit une file d'attente sans limites.

Si le **nom** est spécifié, le VI cherche d'abord une file d'attente existante portant le même nom et retourne son refnum si elle existe. Si aucune file d'attente portant ce nom n'existe et que l'entrée **retourner existants** est Faux (FALSE), le VI crée une nouvelle file d'attente et retourne son refnum. La sortie **nouvellement créé** retourne Vrai (TRUE) si le VI crée une nouvelle file d'attente.

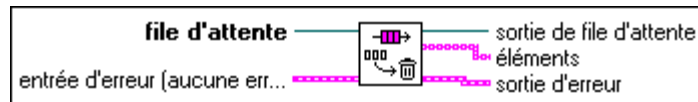
Détruire une file d'attente (Destroy Queue.vi)

Détruit la file d'attente spécifiée et retourne tous ces éléments. Tous les VIs "Insérer un élément de file d'attente" et "Supprimer un élément de file d'attente" qui attendent sur cette file d'attente sont immédiatement suspendus et retournent une erreur.



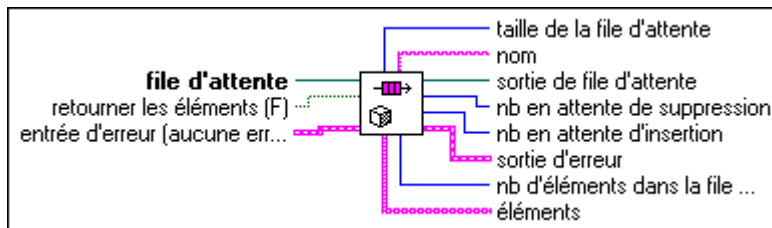
Vider une file d'attente (Flush Queue.vi)

Supprime tous les **éléments** d'une **file d'attente**.



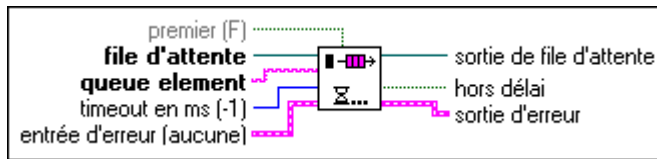
Obtenir l'état d'une file d'attente (Get Queue Status.vi)

Retourne des informations sur l'état courant d'une **file d'attente**.



Insérer un élément de file d'attente (Insert Queue Element.vi)

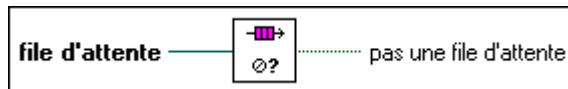
Insère un élément dans une file d'attente.



Le paramètre **à partir du début** spécifie si l'élément est inséré à la fin (par défaut) ou au début de la file d'attente. Si la file d'attente est comblée, le VI attend pendant une durée égale à **timeout en millisecondes** (–1 par défaut ou indéfini) avant de suspendre l'exécution. Si de l'espace devient disponible pendant l'attente, l'élément est inséré et **hors délai** retourne Faux (FALSE). Si la file d'attente reste comblée ou pas valide, **hors délai** retourne Vrai (TRUE).

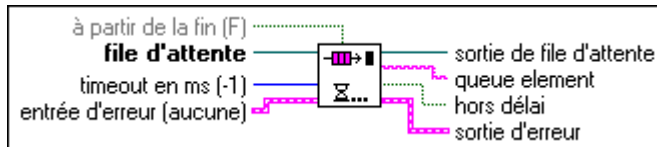
Pas une file d'attente (Not a Queue.vi)

Retourne Vrai (TRUE) si **file d'attente** n'est pas un refnum de file d'attente valide.



Supprimer un élément de file d'attente (Remove Queue Element.vi)

Supprime un élément d'une file d'attente.

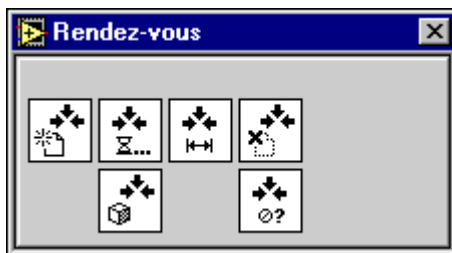


Le paramètre **à partir de la fin** spécifie si l'élément supprimé provient du début (par défaut) ou de la fin de la file d'attente. Si la file d'attente est vide, le VI attend pendant une durée égale à **timeout en millisecondes** (–1 par défaut ou indéfini) avant de suspendre l'exécution. Si un élément devient disponible pendant l'attente, l'élément est retourné et **hors délai** retourne Faux (FALSE). Si aucun élément ne devient disponible ou si la file d'attente n'est pas valide, **hors délai** retourne Vrai (TRUE).

Vis Rendez-vous

Vous pouvez utiliser les VIs Rendez-vous pour synchroniser deux tâches parallèles ou plus à des points spécifiques de l'exécution. Chaque tâche qui atteint le rendez-vous attend jusqu'à ce qu'un nombre spécifié de tâches soient elles aussi en attente avant que toutes les tâches s'exécutent.

Vous pouvez accéder aux VIs Rendez-vous en sélectionnant **Fonctions»Avancé»Synchronisation»Rendez-vous**.



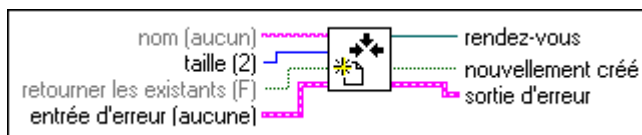
Les VIs Rendez-vous utilisent la commande du **refnum du rendez-vous** située dans la palette **Commandes»Chemin & Refnum**.



Vous pouvez utiliser le refnum du rendez-vous avec les VIs suivants.

Créer un rendez-vous (Create Rendezvous.vi)

Cherchez un **rendez-vous** existant ou créez un nouveau **rendez-vous** et retournez un refnum que vous pouvez utiliser quand vous appelez d'autres VIs Rendez-vous.



La **taille** spécifie combien de tâches doivent atteindre le **rendez-vous** avant de continuer l'exécution. La taille par défaut est 2.

Si le **nom** est spécifié, le VI cherche d'abord un **rendez-vous** existant portant le même nom et retourne son refnum s'il existe. Si aucun rendez-vous portant ce nom n'existe et que l'entrée **retourner existants** est Faux (FALSE), le VI crée un nouveau **rendez-vous** et retourne son refnum. La sortie **nouvellement créé** retourne Vrai (TRUE) si le VI crée un nouveau **rendez-vous**.

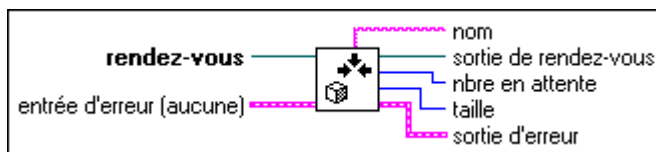
Détruire un rendez-vous (Destroy Rendezvous.vi)

Détruit le rendez-vous spécifié. Tous les VIs qui attendent le rendez-vous sont immédiatement suspendus et retournent une erreur.



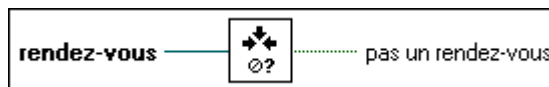
Obtenir l'état d'un rendez-vous (Get Rendezvous Status.vi)

Retourne des informations sur l'état courant d'un **rendez-vous**.



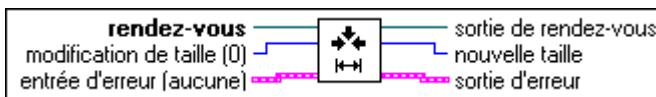
Pas un rendez-vous (Not a Rendezvous.vi)

Retourne Vrai (TRUE) si **rendez-vous** n'est pas un refnum de rendez-vous valide.



Redimensionner un rendez-vous (Resize Rendezvous.vi)

Change la taille d'un **rendez-vous** avec **modification de taille** et retourne **nouvelle taille**.



Si le nombre de tâches qui attendent le **rendez-vous** est inférieur ou égal à **nouvelle taille**, les tâches de la première taille arrêtent d'attendre et continuent l'exécution.

Attendre un rendez-vous (Wait at Rendezvous.vi)

Attend qu'il y ait suffisamment de tâches au rendez-vous.



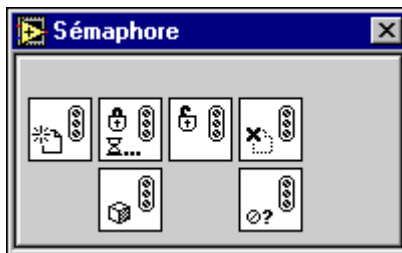
Si le nombre de tâches, y compris la nouvelle, qui attendent au **rendez-vous** est inférieur à la taille du rendez-vous, le VI attend pendant une durée égale à **timeout en millisecondes** (–1 par défaut ou indéfini) avant de suspendre l'exécution. Si suffisamment de tâches arrivent au rendez-vous pendant l'attente, **hors délai** retourne Faux (FALSE). S'il n'y a pas assez de tâches ou que le rendez-vous n'est pas valide, **hors délai** retourne Vrai (TRUE).

Vis Sémaphore

Les sémaphores, également connus sous le nom de Mutex, sont utilisés pour limiter le nombre de tâches exécutées simultanément sur une ressource partagée (protégée). Une ressource protégée ou une section critique peut comprendre l'écriture de variables globales ou la communication avec des instruments externes.

Vous pouvez utiliser les VIs Sémaphore pour synchroniser deux tâches ou plus séparées et parallèles de façon à ce qu'une seule tâche à la fois exécute une section critique de code protégée par un sémaphore commun. Vous pouvez en particulier utiliser ces VIs quand vous souhaitez que d'autres VIs ou parties d'un diagramme attendent qu'un autre VI ou partie d'un diagramme finisse l'exécution d'une section critique.

Vous pouvez accéder aux VIs Sémaphore en sélectionnant **Fonctions»Avancé»Synchronisation»Sémaphore**.



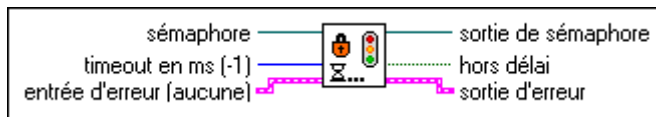
Les VIs Sémaphore utilisent une commande de **refnum de sémaphore** située dans la palette **Commandes»Chemin & Refnum**.



Le refnum de sémaphore peut être utilisé avec les VIs suivants.

Acquérir un sémaphore (Acquire Semaphore.vi)

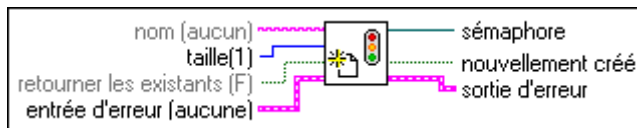
Acquiert l'accès à un sémaphore.



Si le sémaphore est déjà acquis par le nombre maximal de tâches, le VI attend pendant une durée égale à **timeout en millisecondes** (–1 par défaut ou indéfini) avant de suspendre l'opération. Si un sémaphore devient disponible pendant l'attente, **hors délai** retourne Faux (FALSE). Si aucun sémaphore ne devient disponible ou que le sémaphore n'est pas valide, **hors délai** retourne Vrai (TRUE).

Créer un sémaphore (Create Semaphore.vi)

Recherche un sémaphore existant ou crée un nouveau sémaphore et retourne un refnum que vous pouvez utiliser quand vous appelez un autre VI Sémaphore.

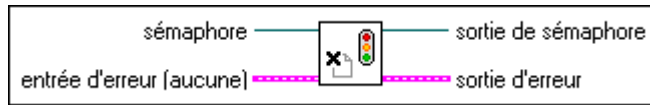


La **taille** spécifie combien de tâches peuvent acquérir un sémaphore en même temps. La taille par défaut est 1.

Si le **nom** est spécifié, le VI cherche d'abord un sémaphore existant portant le même nom et retourne son refnum s'il existe. Si aucun sémaphore portant ce nom n'existe et que l'entrée **retourner existants** est Faux (FALSE), le VI crée un nouveau sémaphore et retourne son refnum. La sortie **nouvellement créé** retourne Vrai (TRUE) si le VI crée un nouveau sémaphore.

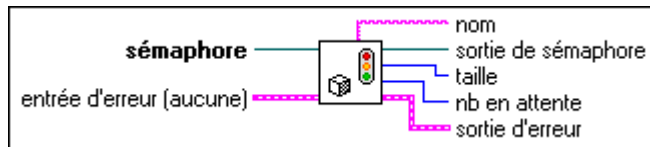
Détruire un sémaphore (Destroy Semaphore.vi)

Détruit le sémaphore spécifié. L'exécution de tous les VIs qui attendent ce sémaphore est immédiatement suspendue et une erreur est retournée.



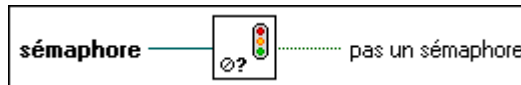
Obtenir l'état d'un sémaphore (Get Semaphore Status.vi)

Retourne des informations sur l'état courant d'un sémaphore.



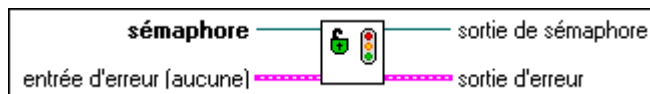
Pas un sémaphore (Not a Semaphore.vi)

Retourne Vrai (TRUE) si **sémaphore** n'est pas un refnum de sémaphore valide.



Libérer un sémaphore (Release Semaphore.vi)

Libère l'accès à un sémaphore.



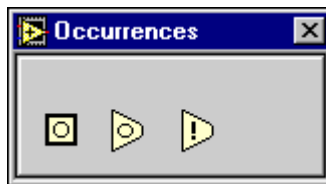
Si un VI attend ce sémaphore, il arrête d'attendre et continue son exécution. Si vous appelez le VI "Libérer un sémaphore" pour un sémaphore que vous n'avez pas acquis, vous augmentez la taille du sémaphore.

Descriptions des fonctions d'occurrences

Vous pouvez utiliser les fonctions d'occurrences pour contrôler des activités séparées et synchronisées. Vous pouvez en particulier utiliser ces fonctions quand vous souhaitez qu'un VI ou qu'une partie du diagramme attende qu'un autre VI ou partie d'un diagramme finisse une tâche sans exiger d'interrogation de la part de LabVIEW.

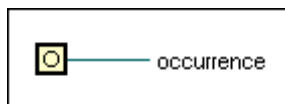
Vous pouvez exécuter la même tâche en utilisant des variables globales, où une boucle interroge la valeur de la variable globale jusqu'à ce que la valeur change. Cependant, cette méthode implique des délais supplémentaires, car la boucle qui interroge la variable globale requiert un temps d'exécution. Avec les occurrences, la boucle d'interrogation est remplacée par une fonction "Attendre une occurrence" et n'exige pas de temps processeur. Quand un diagramme définit une occurrence, LabVIEW active toutes les fonctions "Attendre une occurrence" dans tout diagramme qui attend l'occurrence spécifiée.

L'illustration suivante montre les options disponibles dans la sous-palette **Occurrences**.



Générer une occurrence

Créez une **occurrence** que vous pouvez transmettre aux fonctions "Attendre une occurrence" et "Définir une occurrence".



Normalement, un seul nœud de "Générer une occurrence" est connecté à un ensemble de fonctions "Attendre une occurrence" et "Définir une occurrence". Vous pouvez connecter une fonction "Générer une occurrence" à n'importe quel nombre de fonctions "Attendre une occurrence" et "Définir une occurrence". Il n'est pas nécessaire d'avoir le même nombre de fonctions "Attendre une occurrence" et "Définir une occurrence".

A l'opposé des autres VIs de synchronisation, chaque fonction "Générer une occurrence" dans le diagramme représente une seule et unique occurrence. La fonction "Générer une occurrence" peut donc être considérée comme une constante. Chaque fois qu'une fonction "Générer une occurrence" s'exécute dans un VI, le nœud produit la même valeur. Par exemple, si vous placez une fonction "Générer une occurrence" dans une boucle, la valeur qu'elle produit est la même pour chaque itération de la boucle. En revanche, si vous placez une

fonction “Générer une occurrence” dans le diagramme d’un VI ré-entrant, elle produit une valeur différente pour chaque appelant.

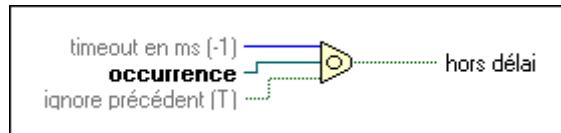
Définir une occurrence

Déclenche l’**occurrence** spécifiée. Tous les diagrammes qui attendent cette occurrence arrêtent d’attendre.



Attendre une occurrence

Attend que la fonction “Définir une occurrence” définisse ou déclenche une **occurrence** donnée.



Partie II

VIs d'acquisition de données

La Partie II introduit l'ensemble des VIs qui fonctionnent avec vos périphériques matériels d'acquisition de données (DAQ). Cette partie contient les chapitres suivants :

- Le chapitre 14, *Introduction aux VIs d'acquisition de données LabVIEW*, contient les informations de base concernant les VIs d'acquisitions de données (DAQ) et vous explique où les trouver dans LabVIEW.
- Le chapitre 15, *VIs d'entrées analogiques simples*, décrit les VIs utilisés pour effectuer des opérations d'entrées analogiques simples.
- Le chapitre 16, *VIs d'entrées analogiques intermédiaires*, décrit les VIs d'entrées analogiques intermédiaires.
- Le chapitre 17, *VIs Utilitaires d'entrées analogiques*, décrit les VIs Utilitaires d'entrées analogiques. Ces VIs, "AI-Lire un balayage" (AI Read One Scan.vi), "AI- Balayer un signal" (AI Waveform Scan.vi) et "AI-Balayer en continu" (AI Continuous Scan.vi), sont des solutions comportant un seul VI apportées aux problèmes classiques d'entrées analogiques. Les VIs Utilitaires d'entrées analogiques sont des VIs de niveau intermédiaire qui dépendent des VIs de niveau avancé.
- Le chapitre 18, *VIs d'entrées analogiques avancées*, contient les descriptions des références des VIs d'entrées analogiques avancées. Ces VIs constituent une interface avec le logiciel NI-DAQ et la base des VIs d'entrées analogiques simples, utilitaires et intermédiaires.
- Le chapitre 19, *VIs de sorties analogiques simples*, décrit les VIs de sorties analogiques simples de LabVIEW qui sont utilisés pour effectuer des opérations de sorties analogiques simples.

- Le chapitre 20, *Vis de sorties analogiques intermédiaires*, décrit les VIs de sorties analogiques intermédiaires. Ces VIs, “AO-Ecrire une mise à jour” (AO Write One Update.vi), “AO-Générer des signaux” (AO Waveform Gen.vi) et “AO-Générer en continu” (AO Continuous Gen.vi), sont des solutions comportant un seul VI apportées à des problèmes classiques de sorties analogiques.
- Le chapitre 21, *Vis Utilitaires de sorties analogiques*, décrit les VIs Utilitaires de sorties analogiques. Ces VIs, “AO-Générer en continu” (AO Continuous Generation.vi) et “AO-Ecrire une mise à jour” (AO Write One Update.vi), sont des solutions comportant un seul VI apportées à des problèmes classiques de sorties analogiques. Les VIs Utilitaires de sorties analogiques sont des VIs de niveau intermédiaire qui dépendent des VIs de niveau avancé.
- Le chapitre 22, *Vis de sorties analogiques avancés*, contient des descriptions de références des VIs de sorties analogiques avancées. Ces VIs constituent une interface avec le logiciel NI-DAQ et la base des VIs de sorties analogiques simples, utilitaires et intermédiaires.
- Le chapitre 23, *Vis d’E/S numériques simples*, décrit les VIs d’E/S numériques simples qui sont utilisés pour exécuter des opérations d’E/S numériques simples.
- Le chapitre 24, *Vis d’E/S numériques intermédiaires*, décrit les VIs E/S numériques intermédiaires. Ces VIs sont des solutions comportant un seul VI apportées à des problèmes numériques classiques.
- Le chapitre 25, *Vis d’E/S numériques avancés*, décrit les VIs d’E/S numériques avancées, qui comprennent les VIs de port et de groupe numérique. Vous utilisez les VIs de port numérique pour des lectures et des écritures immédiates sur des lignes et des ports numériques. Vous pouvez utiliser les VIs de groupe numérique pour des E/S immédiates, à handshaking ou séquencées pour des ports multiples. Ces VIs constituent une interface avec le logiciel NI-DAQ et la base des VIs d’E/S numériques simples et intermédiaires.
- Le chapitre 26, *Vis Compteurs simples*, décrit les VIs Compteurs simples qui sont utilisés dans des opérations de comptage simples.

- Le chapitre 27, *Vis Compteurs intermédiaires*, décrit les VIs Compteurs intermédiaires que vous pouvez utiliser pour programmer des compteurs sur MIO, TIO et d'autres périphériques avec les circuits de compteur DAQ-STC ou Am9513. Ces VIs appellent les VIs Compteurs avancés pour configurer les compteurs propres aux opérations classiques et démarrer, lire et arrêter les compteurs. Vous pouvez configurer ces VIs pour générer des impulsions uniques ou des séries d'impulsions continues pour compter des événements et du temps écoulé, pour diviser un signal et les VIs Compteurs simples appellent les VIs Compteurs intermédiaires pour plusieurs opérations de génération d'impulsion, de décompte et de mesure.
- Le chapitre 28, *Vis Compteurs avancés*, décrit les VIs qui configurent et contrôlent les compteurs matériels. Vous pouvez utiliser ces VIs pour générer des signaux carrés avec des facteurs d'utilisation variables, pour compter des événements et pour mesurer des périodes et des fréquences.
- Le chapitre 29, *Vis d'étalonnage et de configuration*, décrit les VIs utilisés pour étalonner des périphériques spécifiques, définir et retourner des informations d'étalonnage.
- Le chapitre 30, *Vis de conditionnement de signaux*, décrit les VIs de conditionnement de signaux d'acquisition de données que vous pouvez utiliser pour convertir les tensions d'entrées analogiques lues à partir de détecteurs de température à résistance (RTD), de jauges de contrainte ou de thermocouples en unités de contrainte ou de température.

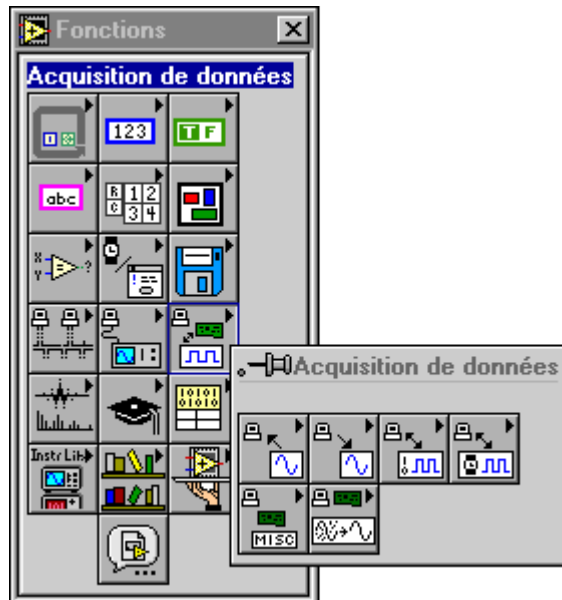
Introduction aux VIs d'acquisition de données LabVIEW

Ce chapitre présente des informations de base sur les VIs Acquisition de données (DAQ) et indique où vous pouvez les trouver dans LabVIEW. Les descriptions de ces VIs figurent dans les chapitres 14 à 29.

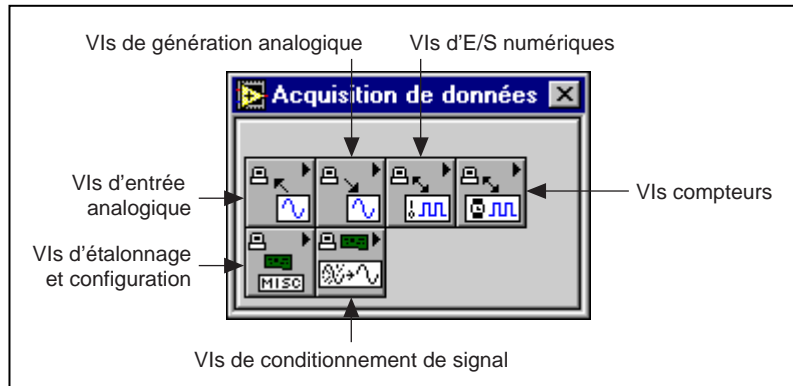
LabVIEW dispose d'une collection de VIs qui fonctionnent avec vos périphériques DAQ et avec lesquels vous pouvez développer des applications d'acquisition et de contrôle.

Vous trouverez les VIs DAQ de la palette **Fonctions** de votre diagramme dans LabVIEW. Les VIs DAQ se trouvent en bas de la palette **Fonctions**.

Pour ouvrir la palette **Acquisition de Données**, sélectionnez **Fonctions»Acquisition de données**, comme indiqué ci-après.



La palette **Acquisition de données** comprend six icônes de sous-palettes pour accéder aux différentes classes de VIs DAQ. L'illustration ci-après explique la signification de chaque icône de la palette **Acquisition de données**.



La disposition de cette partie du manuel suit l'ordre d'apparition des icônes des VIs DAQ de la palette **Acquisition de données** de gauche à droite. Par exemple, dans cette section, les chapitres sur les VIs d'entrées analogiques sont suivis par les chapitres sur les VIs de sorties analogiques, puis par les chapitres sur les E/S numériques, et ainsi de suite. Le plus souvent, il existe plusieurs chapitres sur une classe de VIs DAQ d'une même palette, car plusieurs palettes de VI comportent des sous-palettes.

Aide en ligne pour les VIs DAQ

Vous trouverez des informations en ligne utiles sur des VIs individuels grâce à la fenêtre d'aide de LabVIEW (**Aide** » **Visualiser l'aide**). Quand vous placez le curseur sur l'icône d'un VI, le diagramme de câblage et les noms des paramètres de ce VI apparaissent dans la fenêtre d'aide. Vous pouvez également trouver des informations sur les commandes et les indicateurs de la face-avant en positionnant le curseur sur la commande ou l'indicateur quand la fenêtre d'aide est ouverte. Pour plus d'informations sur la fenêtre d'aide de LabVIEW, consultez la section *Obtention d'aide* dans le chapitre 1, *Introduction à la programmation en G*, dans le *Manuel de référence de programmation en G*.

En plus de la fenêtre d'aide, LabVIEW dispose d'informations en ligne plus détaillées. Pour accéder à ces informations, sélectionnez **Aide»Référence en ligne**. Pour la plupart des objets du diagramme, vous pouvez sélectionner **Référence en ligne** dans le menu local de l'objet pour accéder à la description en ligne. Vous pouvez également accéder à ces informations en appuyant sur le bouton situé sur la gauche, en bas de la fenêtre d'aide de LabVIEW. Pour des informations sur la création de vos propres fichiers de référence en ligne, reportez-vous à la section *Création de vos propres fichiers d'aide* dans le chapitre 5, *Impression et documentation des VIs*, du *Manuel de référence de programmation en G*.

**Remarque**

N'utilisez que les entrées dont vous avez réellement besoin pour chaque VI. LabVIEW règle toutes les entrées non câblées sur leurs valeurs par défaut. Une grande partie des entrées de fonctions DAQ sont optionnelles et n'apparaissent pas dans la fenêtre d'Aide pour diagramme simple. En général, ces entrées spécifient des options rarement utilisées. Si une entrée est nécessaire, votre câblage de VI reste "brisé" jusqu'à ce qu'une valeur soit connectée à l'entrée. Les entrées nécessaires apparaissent en caractères gras dans la fenêtre Aide, les recommandées en caractères normaux et les optionnelles en caractères gris. Les valeurs par défaut des entrées apparaissent entre parenthèses à côté du nom de l'entrée dans la fenêtre d'Aide.

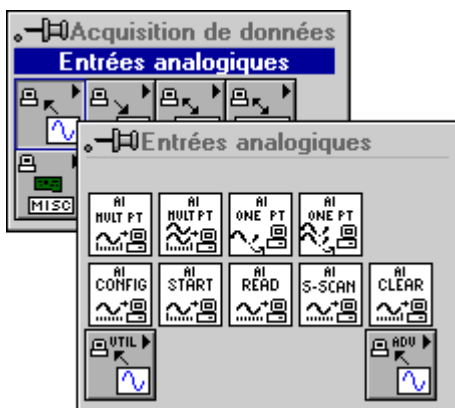
**Remarque**

Quelques VIs DAQ utilisent un type de données énuméré comme terminal de commande ou d'indicateur. Si vous connectez une valeur numérique à un indicateur énuméré, LabVIEW associe ce nombre à l'élément d'énumération le plus proche. Si vous connectez une commande énumérée à une valeur numérique, cette valeur est égale à l'indice d'énumération.

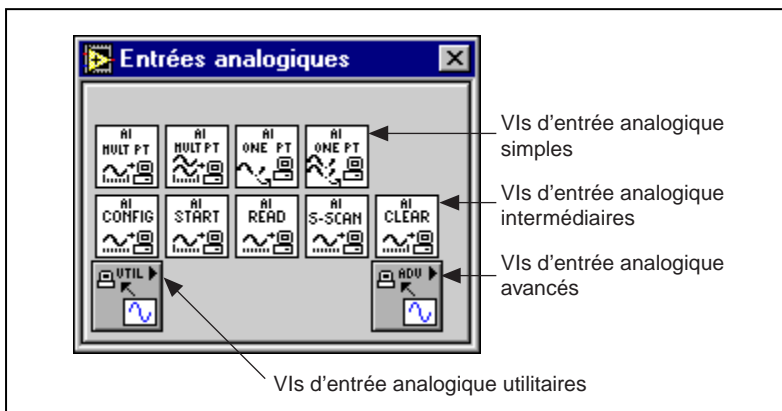
VIs d'entrées analogiques

Ces VIs réalisent des opérations d'entrées analogiques.

Vous pouvez trouver les VIs d'entrées analogiques en sélectionnant **Fonctions»Acquisition de données»Entrées analogiques**. Quand vous cliquez sur l'icône d'entrée analogique de la palette **Acquisition de données**, la palette **Entrées analogiques** s'affiche comme illustré ci-dessous.



Il existe quatre classes de VIs d'entrées analogiques de la palette **Entrées analogiques**. Les VIs d'entrées analogiques simples, les VIs d'entrées analogiques intermédiaires, les VIs Utilitaires d'entrées analogiques et les VIs d'entrées analogiques avancés. L'illustration ci-dessous présente ces classes de VIs.



VIs d'entrées analogiques simples

Les VIs d'entrées analogiques simples réalisent des opérations d'entrées analogiques simples. Vous pouvez exécuter ces VIs à partir de la face-avant ou comme des sous-VIs dans des applications simples.

Vous pouvez utiliser chaque VI séparément pour exécuter une opération analogique simple. A la différence des VIs de niveaux intermédiaire et avancé, les VIs d'entrées analogiques simples indiquent automatiquement si une erreur s'est produite à l'aide d'une boîte de dialogue qui vous demande d'arrêter l'exécution du VI ou d'ignorer l'erreur.

Les VIs d'entrées analogiques simples représentent une interface facile et simple ne comportant que les entrées et les sorties les plus communes. Pour des applications plus complexes et de meilleures performances, utilisez les VIs d'entrées analogiques de niveau intermédiaire ou les VIs d'entrées analogiques de niveau avancé.

Consultez le chapitre 15, *VIs d'entrées analogiques simples*, pour des informations spécifiques sur ces VIs.

VIs d'entrées analogiques intermédiaires

Vous trouverez des VIs d'entrées analogiques de niveau intermédiaire à deux endroits différents de la palette **Entrées analogiques**. Vous les trouverez dans la seconde rangée de la palette **Entrées analogiques** et de la palette **Utilitaires des entrées analogiques**, dont la description figure plus loin dans ce texte. La construction des VIs d'entrées analogiques intermédiaires, AI-Configurer (AI-Config.vi), AI-Démarrer (AI-Start.vi), AI-Lire (AI-Read.vi), AI-Balayage Unique (AI-SingleScan.vi) et AI-Réinitialiser (AI-Clear.vi), repose sur la couche de construction fondamentale, constituée par les VIs d'entrées analogiques avancés. Ces VIs sont presque aussi puissants que les VIs de niveau avancé et les regroupent en une séquence logique et nette.

Consultez le chapitre 16, *VIs d'entrées analogiques intermédiaires*, pour des informations spécifiques sur ces VIs.

Les VIs Utilitaires d'entrées analogiques



icône utilitaire des entrées analogiques

Ouvrez la palette **Utilitaires des entrées analogiques** en sélectionnant l'icône Utilitaire des entrées analogiques de la palette **Entrées analogiques**. Les VIs Utilitaires des entrées analogiques, AI-Lire un balayage (AI Read Scan.vi), AI-Balayer un signal (AI Waveform Scan.vi) et AI-Balayage continu (AI Continuous Scan.vi), sont des solutions à VI unique pour résoudre des problèmes classiques d'entrées analogiques. Ces VIs sont simples mais peu flexibles. La construction de ces trois VIs repose sur les VIs d'entrées analogiques intermédiaires de la palette **Entrées analogiques**.

Consultez le chapitre 17, *VIs Utilitaires d'entrées analogiques*, pour des informations spécifiques sur ces VIs.

VIs d'entrées analogiques avancés



icône d'entrées analogiques avancées

Ouvrez la palette **Entrées analogiques avancées** en sélectionnant l'icône présentée ci-contre de la palette **Entrées analogiques**. Ces VIs constituent une interface avec le logiciel d'acquisition de données NI-DAQ et sont le fondement des VIs d'entrées analogiques simples, utilitaires et intermédiaires.

Consultez le chapitre 18, *VIs d'entrées analogiques avancés*, pour des informations spécifiques sur les VIs.

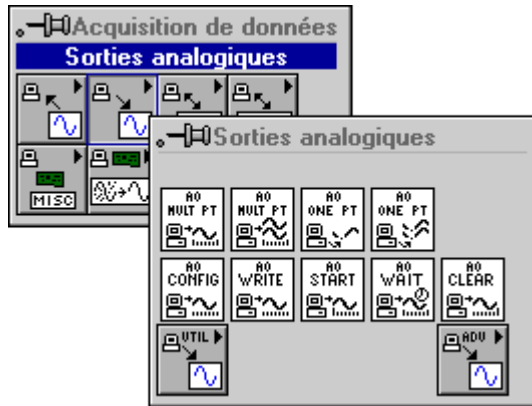
Exemples de VIs concernant les entrées analogiques

Pour des exemples sur l'emploi des VIs d'entrées analogiques, reportez-vous à la bibliothèque `examples\daq\anlogin\anlogin.llb`.

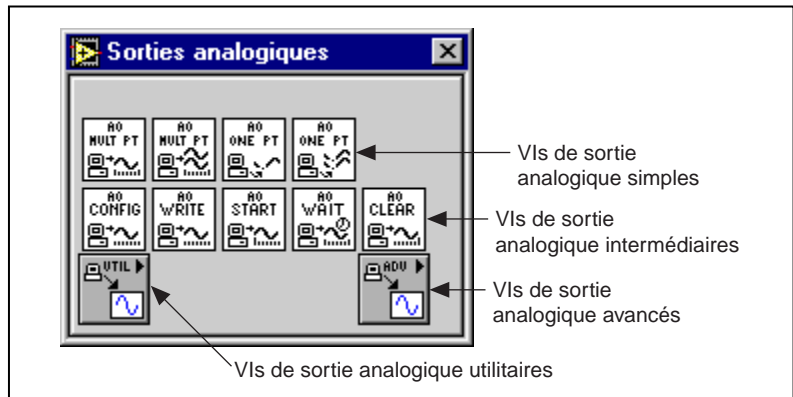
VIs de sorties analogiques

Ces VIs réalisent des opérations de sorties analogiques.

Trouvez les VIs de sorties analogiques en sélectionnant **Fonctions»Acquisition de données»Sorties analogiques**. Quand vous cliquez sur l'icône de sorties analogiques de la palette **Acquisition de données**, la palette **Sorties analogiques** s'affiche, comme illustré ci-dessous.



Il existe quatre classes de VIs de sorties analogiques de la palette **Sorties analogiques** : les VIs de sorties analogiques simples, les VIs de sorties analogiques intermédiaires, les VIs Utilitaires de sorties analogiques et les VIs de sorties analogiques avancés. L'illustration suivante présente ces classes de VIs.



VIs de sorties analogiques simples

Les VIs de sorties analogiques simples réalisent des opérations simples de sorties analogiques. Vous pouvez exécuter ces VIs à partir de leur face-avant ou comme des sous-VIs dans des applications simples.

Vous pouvez utiliser chaque VI séparément pour exécuter une opération de sortie analogique simple. A la différence des VIs de niveaux intermédiaire et avancé, les VIs de sorties analogiques simples indiquent automatiquement si une erreur s'est produite pendant l'exécution du VI à

l'aide d'une boîte de dialogue qui vous demande d'arrêter l'exécution du VI ou d'ignorer l'erreur.

Les VIs de sorties analogiques simples représentent une interface facile et simple ne comportant que les entrées et les sorties les plus communes. Pour des applications plus complexes et de meilleures performances, utilisez les VIs de sorties analogiques de niveau intermédiaire ou les VIs de sorties analogiques de niveau avancé.

Consultez le chapitre 19, *VIs de sorties analogiques simples*, pour des informations spécifiques sur ces VIs.

VIs de sorties analogiques intermédiaires

Vous trouverez des VIs de sorties analogiques de niveau intermédiaire à deux endroits différents de la palette **Sorties analogiques**. Vous les trouverez dans la seconde rangée de la palette **Sorties analogiques**. Les autres VIs de niveau intermédiaire sont dans la palette **Utilitaires des sorties analogiques**, dont la description figure plus loin dans ce texte. La construction des VIs de sorties analogiques intermédiaires, AO-Configurer (AO Config.vi), AO-Démarrer (AO Start.vi), AO-Lire (AO Read.vi), AO-BalayageUnique (AO SingleScan.vi) et AO-Réinitialiser (AO Clear.vi), repose sur la couche de construction fondamentale, constituée par les VIs de sorties analogiques avancés. Ces VIs sont presque aussi puissants que les VIs de niveau avancé et les regroupent en une séquence logique et nette.

Consultez le chapitre 20, *VIs de sorties analogiques intermédiaires*, pour des informations spécifiques sur ces VIs.

VIs Utilitaires de sorties analogiques



icône utilitaire des sorties analogiques

Ouvrez la palette **Utilitaires des sorties analogiques** en sélectionnant l'icône Utilitaire des sorties analogiques de la palette **Sorties analogiques**. Les VIs Utilitaires de sorties analogiques, AO-Lire un balayage (AO Read Scan.vi), AO-Générer un signal (AO Generate a Waveform.vi) et AO-Générer en continu (AO Continuous Gen.vi), sont des solutions à des problèmes de sorties analogiques classiques. Ces VIs sont simples mais peu flexibles. La construction de ces trois VIs repose sur les VIs de sorties analogiques intermédiaires de la palette **Sorties analogiques**.

Consultez le chapitre 21, *VIs Utilitaires de sorties analogiques*, pour des informations spécifiques sur ces VIs.

VIs de sorties analogiques avancés



icône des sorties analogiques avancées

Ouvrez la palette **Sorties analogiques avancées** en sélectionnant l'icône des sorties analogiques avancées de la palette **Sorties analogiques**. Ces VIs constituent l'interface avec le logiciel d'acquisition de données NI-DAQ et sont le fondement des VIs de sorties analogiques simples, utilitaires et intermédiaires.

Etant donné que tous ces VIs dépendent des VIs de niveau avancé, vous pouvez consulter le chapitre 22, *VIs de sorties analogiques avancées*, pour plus d'informations sur les entrées, les sorties et leur fonctionnement.

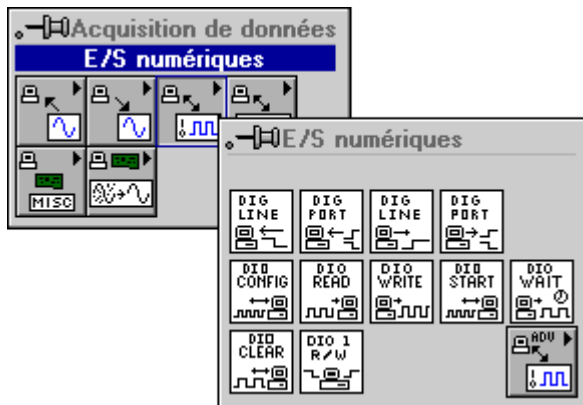
Exemples de VIs concernant les sorties analogiques

Pour des exemples sur l'emploi des VIs de sorties analogiques, reportez-vous à la bibliothèque `examples\daq\anlogout\anlogout.llb`.

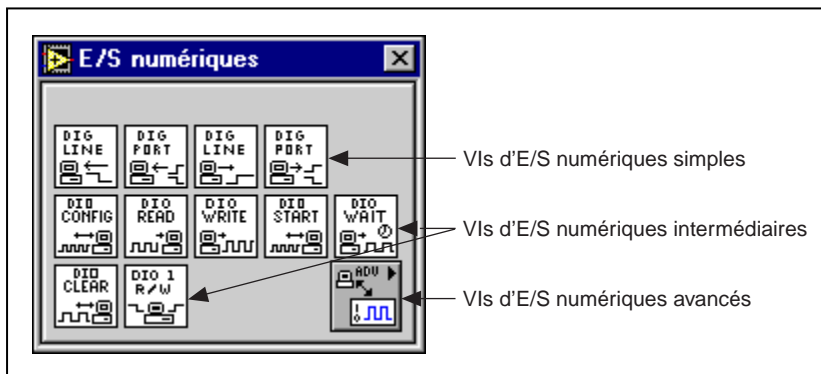
VIs de fonctions numériques

Ces VIs réalisent des opérations d'entrées/sorties numériques.

Recherchez les VIs d'E/S numériques en sélectionnant **Fonctions» Acquisition de données» E/S numériques**. Quand vous cliquez sur l'icône d'E/S numériques de la palette **Acquisition de données**, la palette **E/S numériques** s'affiche, comme illustré ci-dessous.



Il existe trois classes de VIs d'E/S numériques de la palette **E/S numériques** : les VIs d'E/S numériques simples, les VIs d'E/S numériques intermédiaires et les VIs d'E/S numériques avancés. L'illustration ci-dessous présente ces classes de VIs.



VIs d'E/S numériques simples

Les VIs d'E/S numériques simples réalisent des opérations numériques simples. Vous pouvez exécuter ces VIs à partir de leur face-avant ou comme des sous-VIs dans des applications simples.

Vous pouvez utiliser chaque VI séparément pour exécuter une opération numérique simple. A la différence des VIs de niveau intermédiaire et avancé, les VIs d'E/S numériques simples indiquent automatiquement si une erreur s'est produite pendant l'exécution du VI à l'aide d'une boîte de dialogue qui vous demande d'arrêter l'exécution du VI ou d'ignorer l'erreur.

Les VIs d'E/S numériques simples sont en fait composés de VIs d'E/S numériques avancés. Les VIs d'E/S numériques simples représentent une interface facile et simple ne comportant que les entrées et les sorties les plus communes. Pour des applications plus complexes et de meilleures performances, utilisez les VIs de niveau intermédiaire ou avancé.

Consultez le chapitre 23, *VIs d'E/S numériques simples*, pour des informations spécifiques sur ces VIs.

VIs d'E/S numériques intermédiaires

Vous trouverez des VIs d'E/S numériques de niveau intermédiaire dans les deuxième et troisième rangées de la palette **E/S numériques**. La construction des VIs d'E/S numériques intermédiaires repose sur la couche

de construction fondamentale, représentée par les VIs d'E/S numériques avancés. Ces VIs sont presque aussi puissants que les VIs de niveau avancé et regroupent les VIs en une séquence logique et nette.

Consultez le chapitre 24, *VIs d'E/S numériques intermédiaires*, pour des informations spécifiques sur ces VIs.

VIs d'E/S numériques avancés



icône d'E/S numériques avancées

Ouvrez la palette **E/S numériques avancées** en sélectionnant l'icône des E/S numériques avancées de la palette **E/S numériques**. Ces VIs constituent l'interface avec le logiciel d'acquisition de données NI-DAQ et sont le fondement des VIs d'E/S numériques simples et intermédiaires.

Etant donné que tous ces VIs dépendent des VIs de niveau avancé, vous pouvez consulter le chapitre 25, *VIs d'E/S numériques avancés*, pour plus d'informations sur les entrées, les sorties et leur fonctionnement.

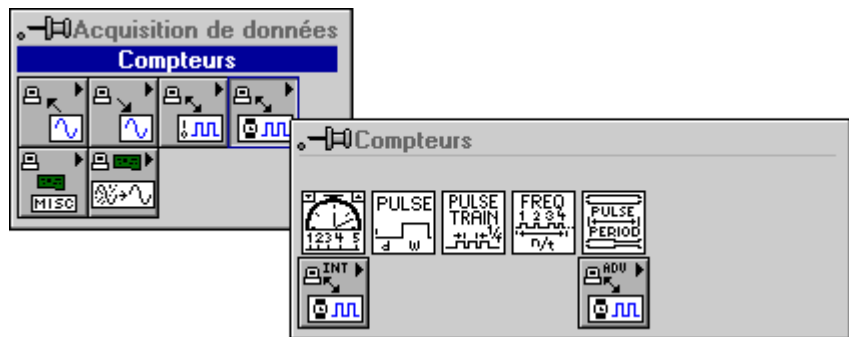
Exemples de VIs d'E/S numériques

Pour des exemples sur l'emploi des VIs d'E/S numériques, reportez-vous à la bibliothèque `examples\daq\digital\digio.lib`.

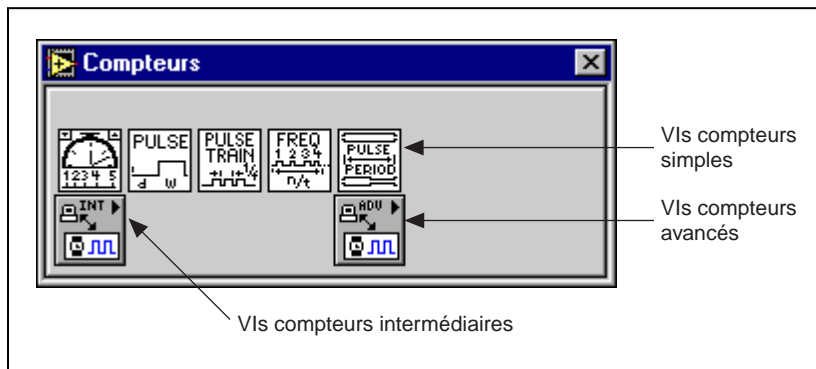
VIs Compteurs

Ces VIs réalisent des opérations de comptage.

Vous pouvez accéder aux VIs Compteurs en sélectionnant la palette **Fonctions»Acquisition de données»Compteurs**. Quand vous cliquez sur l'icône Compteurs de la palette **Acquisition de données**, la palette **Compteurs** s'affiche, comme illustré ci-dessous.



Il existe trois classes de VIs Compteurs de la palette **Compteurs** : les VIs simples, intermédiaires et avancés. L'illustration ci-après présente ces classes de VIs.



VIs Compteurs simples

Les VIs Compteurs simples réalisent des opérations simples de comptage. Vous pouvez exécuter ces VIs à partir de leur face-avant ou comme des sous-VIs dans des applications simples.

Vous pouvez utiliser chaque VI séparément pour exécuter une opération de comptage simple. A la différence des VIs de niveaux intermédiaire et avancé, les VIs Compteurs simples indiquent automatiquement si une erreur s'est produite pendant l'exécution du VI à l'aide d'une boîte de dialogue qui vous demande d'arrêter l'exécution du VI ou d'ignorer l'erreur.

Les VIs Compteurs simples sont en fait composés de VIs Compteurs intermédiaires puis de VIs Compteurs avancés. Les VIs simples représentent une interface facile et simple ne comportant que les entrées et les sorties les plus communes. Pour des applications plus complexes et de meilleures performances, utilisez les VIs de niveau intermédiaire ou avancé.

Consultez le chapitre 26, *VIs Compteurs simples*, pour des informations spécifiques sur ces VIs.

VIs Compteurs intermédiaires



Icône VIs Compteurs intermédiaires

Vous trouverez des VIs Compteurs de niveau intermédiaire dans la seconde rangée de la palette **Compteurs**. La construction de ces VIs repose sur la couche de construction fondamentale, représentée par les VIs Compteurs avancés. Ces VIs sont presque aussi puissants que les VIs de niveau avancé et les regroupent en une séquence logique et nette.

Consultez le chapitre 27, *VIs Compteurs intermédiaires*, pour des informations spécifiques sur ces VIs.

VIs Compteurs avancés



Icône VIs Compteurs avancés

Ouvrez la palette **Compteurs avancés** en sélectionnant l'icône des Compteurs avancés de la palette **Compteurs**. Ces VIs constituent l'interface avec le logiciel d'acquisition de données NI-DAQ et sont le fondement des VIs Compteurs simples et intermédiaires.

Etant donné que tous ces VIs dépendent des VIs de niveau avancé, vous pouvez consulter le chapitre 28, *VIs Compteurs avancés*, pour plus d'informations sur les entrées, les sorties et leur fonctionnement.

Exemples de VIs Compteurs

Pour des exemples sur l'emploi des VIs Compteurs, ouvrez les bibliothèques d'exemples `examples\daq\counter\DAQ-STC.llb`, `examples\daq\counter\am9513.llb`, et `examples\daq\counter\8253.llb`.

VIs d'étalonnage et de configuration

Ces VIs effectuent l'étalonnage de périphériques spécifiques et paramètrent et rendent des informations sur la configuration.

Reportez-vous au chapitre 29, *VIs d'étalonnage et de configuration*, pour des informations sur l'emplacement de ces VIs et des exemples.

VIs de conditionnement de signaux

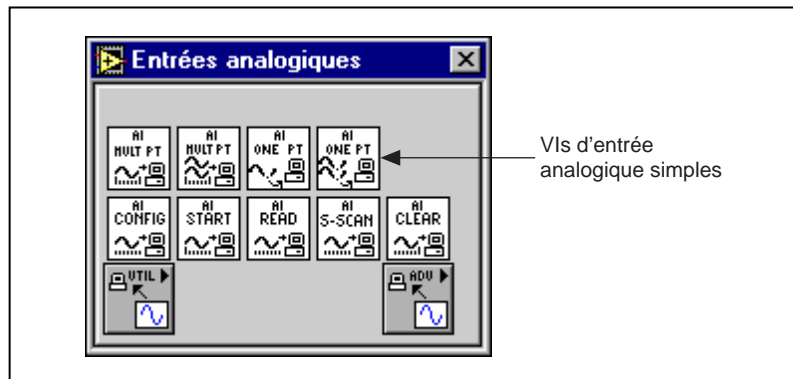
Ces VIs convertissent les tensions d'entrée analogique lues à partir de capteurs de température à résistance (RTD), de jauges de contrainte ou de thermocouples en unités de contrainte ou de température.

Reportez-vous au chapitre 30, *VIs de conditionnement de signaux*, pour des informations sur l'emplacement de ces VIs et des exemples.

Vis d'entrées analogiques simples

Ce chapitre décrit les VIs d'entrées analogiques simples, qui réalisent des opérations simples d'entrées analogiques. Vous pouvez exécuter ces VIs à partir de leur face-avant ou comme des sous-VIs dans des applications de base.

Vous trouverez les VIs d'entrées analogiques dans la palette **Fonctions»Acquisition de données»Entrées analogiques**. Les VIs d'entrées analogiques simples se situent dans la rangée supérieure de la palette **Entrées analogiques**, comme illustré ci-dessous.

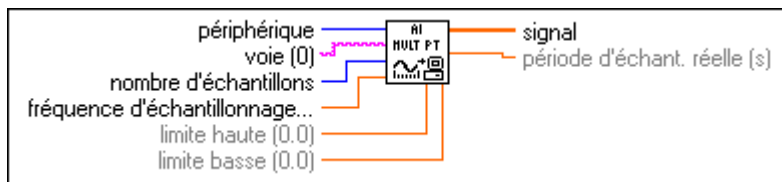


Descriptions des VIs d'entrées analogiques simples

Les VIs d'entrées analogiques simples suivants sont disponibles.

AI-Acquérir un signal (AI Acquire Waveform.vi)

Acquiert un nombre spécifié d'échantillons à une fréquence d'échantillonnage spécifiée à partir d'une voie d'entrée et retourne les données acquises.

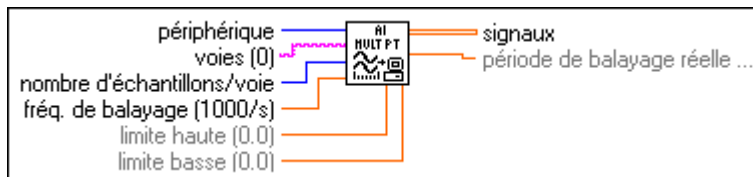


Le VI "AI-Acquérir un signal" (AI Acquire Waveform.vi) effectue une mesure séquentielle par le matériel d'un signal (des lectures à tensions multiples à une fréquence d'échantillonnage spécifiée) sur une voie unique d'entrées analogiques. Si une erreur survient, une boîte de dialogue apparaît, vous offrant le choix d'arrêter le VI ou de continuer.

Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour obtenir les numéros de voies et les limites d'entrées disponibles avec votre périphérique DAQ.

AI-Acquérir des signaux (AI Acquire Waveforms.vi)

Acquiert des données sur les voies spécifiées et échantillonne les voies avec la fréquence de balayage spécifiée.

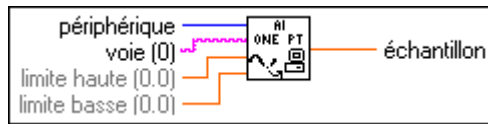


Le VI "AI-Acquérir des signaux" (AI Acquire Waveforms.vi) effectue une mesure séquentielle de plusieurs signaux dans les voies d'entrées analogiques spécifiées. Si une erreur survient, une boîte de dialogue apparaît, vous donnant le choix d'interrompre l'opération ou de continuer l'exécution.

Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour obtenir les numéros de voies et les limites d'entrées disponibles avec votre périphérique DAQ.

AI-Echantillonner une voie (AI Sample Channel.vi)

Mesure le signal connecté à la voie spécifiée et retourne la valeur mesurée.

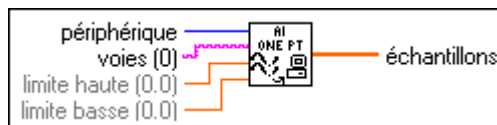


Le VI “AI-Echantillonner une voie” (AI Sample Channel.vi) effectue une mesure unique non séquentielle d’une voie. Si une erreur survient, une boîte de dialogue apparaît, vous offrant le choix d’arrêter le VI ou de continuer.

Reportez-vous à l’Annexe B, *Capacités matérielles DAQ*, pour obtenir les numéros de voies et les limites d’entrées disponibles en fonction de votre périphérique DAQ.

AI-Echantillonner des voies (AI Sample Channels.vi)

Exécute une lecture unique de chaque voie spécifiée.



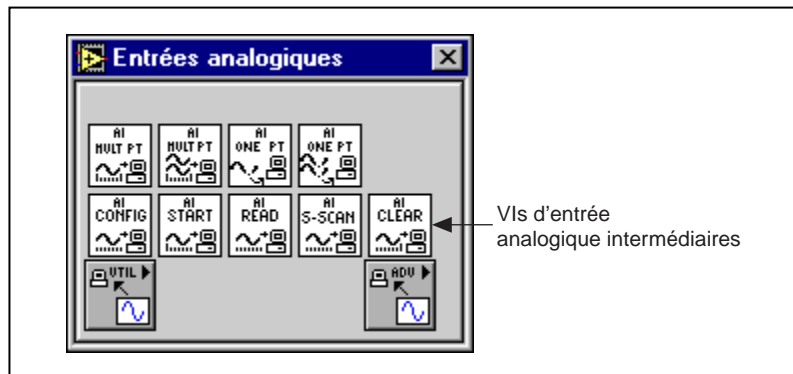
Le VI “AI-Echantillonner des voies” (AI Sample Channels.vi) mesure une valeur unique de chacune des voies d’entrées analogiques spécifiées. Si une erreur survient, une boîte de dialogue apparaît, vous offrant le choix d’arrêter le VI ou de continuer.

Reportez-vous à l’Annexe B, *Capacités matérielles DAQ*, pour obtenir les numéros de voies et les limites d’entrées disponibles avec votre périphérique DAQ.

VIs d'entrées analogiques intermédiaires

Ce chapitre décrit les VIs d'entrées analogiques intermédiaires. Ces VIs sont faciles d'emploi mais peu flexibles.

Vous pouvez accéder aux VIs d'entrées analogiques intermédiaires en sélectionnant la palette **Fonctions»Acquisition de données»Entrées analogiques**. Les VIs d'entrées analogiques intermédiaires sont dans la seconde rangée de la palette **Entrées analogiques**, comme illustré ci-dessous.



Gestion d'erreurs

LabVIEW simplifie la gestion d'erreurs dans les VIs d'entrées analogiques intermédiaires. Chaque VI de niveau intermédiaire comporte un cluster d'entrée **entrée d'erreur** et un cluster de sortie **sortie d'erreur**. Les clusters comprennent un booléen qui indique la présence ou non d'une erreur, le code d'erreur et le nom du VI qui a retourné l'erreur. Si **entrée d'erreur** indique une erreur, le VI transmet les informations concernant l'entrée d'erreur au cluster **sortie d'erreur** et arrête l'exécution.



Remarque *Le VI "AI-Réinitialiser" (AI Clear.vi) est une exception à la règle. En effet, ce VI réinitialise toujours l'acquisition sans savoir si l'entrée indique une erreur.*

Quand vous utilisez n'importe quel VI d'entrées analogiques intermédiaires dans une boucle While, arrêtez la boucle si l'état du cluster **sortie d'erreur** indique Vrai (TRUE). Si vous câblez le cluster d'erreur au VI du Gestionnaire général d'erreurs, ce VI déchiffre les informations relatives à l'erreur et les affiche sous la forme d'une boîte de dialogue.

Le VI "Gestionnaire général d'erreurs" (General Error Handler.vi) se trouve dans la palette **Fonctions»Temps et dialogue** de LabVIEW.

Descriptions des VIs d'entrées analogiques intermédiaires

Les VIs d'entrées analogiques intermédiaires suivants sont disponibles.

AI-Réinitialiser (AI Clear.vi)

Efface la tâche d'entrée analogique associée avec **taskID d'entrée**.



Le VI "AI-Réinitialiser" (AI-Clear.vi) arrête l'acquisition associée avec **taskID d'entrée** et libère les ressources internes associées, y compris les buffers. Avant de commencer une nouvelle acquisition, vous devez appeler le VI "AI-Configurer" (AI Config.vi). Reportez-vous au chapitre 18, *VIs d'entrées analogiques avancés*, pour une description du VI "AI-Contrôler" (AI Control.vi).



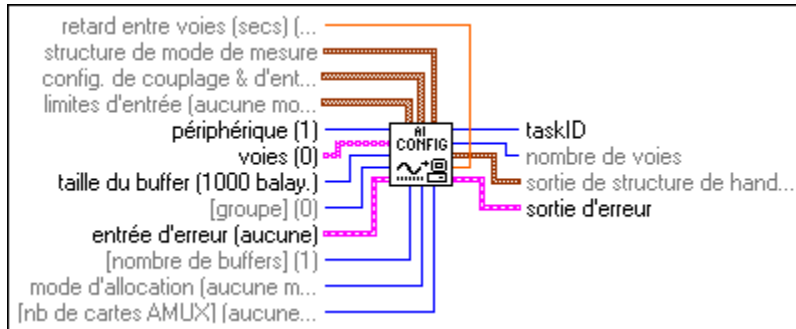
Remarque *Le VI "AI-Réinitialiser" (AI Clear.vi) réinitialise toujours l'acquisition sans savoir si le cluster entrée d'erreur indique une erreur.*

Quand vous utilisez n'importe quel VI d'entrées analogiques intermédiaires dans une boucle While, arrêtez la boucle si le booléen **état** du cluster **sortie d'erreur** est Vrai (TRUE). Si vous câblez le cluster d'erreur au VI du Gestionnaire général d'erreurs, le VI déchiffre les informations relatives à l'entrée d'erreur et les affiche dans une boîte de dialogue.

Le VI du Gestionnaire général d'erreurs se trouve dans la palette **Fonctions»Temps et dialogue** de LabVIEW. Pour plus d'informations sur ce VI, reportez-vous au chapitre 10, *Fonctions de temps, de dialogue et d'erreurs*.

AI-Configurer (AI Config.vi)

Configure une opération d'entrées analogiques pour un ensemble de voies spécifié. Ce VI configure le matériel et attribue un buffer pour une opération d'entrées analogiques dans un buffer.



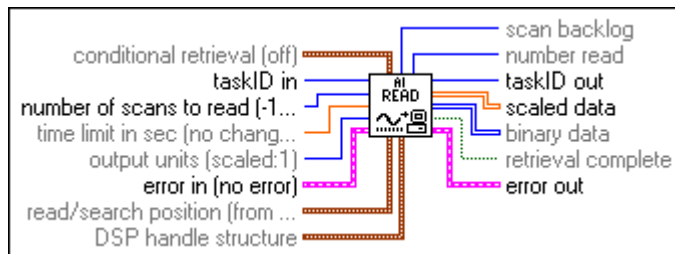
Vous ne pouvez allouer plusieurs buffers qu'avec l'un des périphériques suivants.

- (Macintosh) NB-A2000, NB-A2100 et NB-A2150

Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour obtenir les gammes, les limites d'entrées et l'ordre du balayage des voies disponibles sur votre périphérique DAQ National Instruments.

AI-Lire (AI Read.vi)

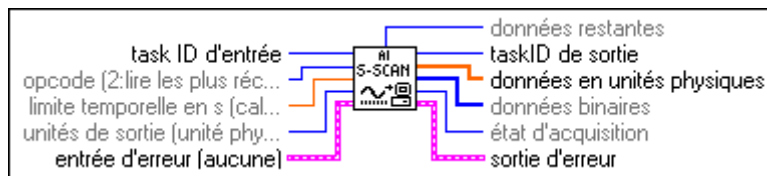
Lit des données d'une acquisition de données bufférisée.



Le VI "AI-Lire" (AI Read.vi) appelle le VI "AI-Lire un buffer" (AI Buffer Read.vi) pour lire des données dans une acquisition d'entrées analogiques bufférisée.

AI-Balayage simple (AI Single Scan.vi)

Retourne un balayage de données d'un groupe de voies précédemment configuré.



Si vous avez déjà commencé une acquisition avec le VI “AI-Démarrer” (AI Start.vi), ce VI lit un balayage de données à partir du buffer d’acquisition ou de la FIFO embarquée si l’acquisition n’est pas bufférisée. Si vous n’avez pas commencé l’acquisition, ce VI la commence, récupère un balayage de données et la termine. La configuration de groupe détermine les voies que le VI échantillonne.

Si vous n’appellez pas le VI “AI-Démarrer” (AI Start.vi), ce VI commence un balayage unique à l’aide de la fréquence d’horloge sûre la plus rapide. Modifiez la fréquence d’horloge avec le VI “AI-Configurer” (AI Config.vi).

Si vous exécutez le VI “AI-Démarrer” (AI Start.vi), un signal d’horloge entame les balayages.

Vous devez utiliser le VI “AI-Démarrer” (AI Start.vi) pour définir une source d’horloge externe, pour des conversions séquencées de façon externe.

Si les sources d’horloge sont internes et que vous n’allouez pas de mémoire, une acquisition séquencée non bufférisée commence quand vous exécutez le VI “AI-Démarrer” (AI Start.vi). Utilisez ce type d’acquisition pour synchroniser des entrées et des sorties analogiques dans une application de contrôle de point par point. Les périphériques suivants ne supportent pas des acquisitions séquencées, non bufférisées.

- (Macintosh) NB-A2000, NB-A2100, et NB-A2150

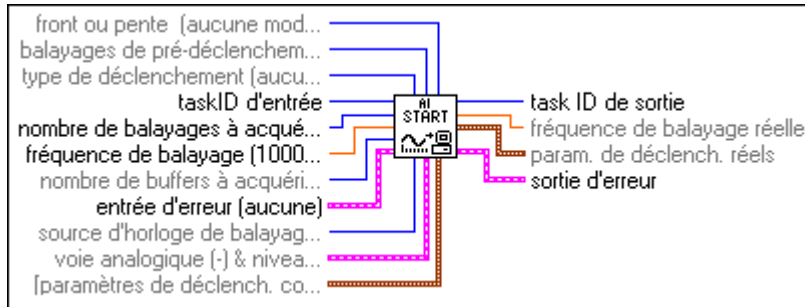


Remarque *LabVIEW redémarre le dispositif si un débordement FIFO survient lors d’une acquisition séquencée non bufférisée.*

Quand vous définissez le paramètre **opcode** sur 1 pour une acquisition non bufférisée, le VI lit un balayage dans la FIFO et retourne les données. Si **opcode** est égal à 2, le VI lit la FIFO jusqu’à ce qu’elle soit vide et retourne le dernier balayage lu.

AI-Démarrer (AI Start.vi)

Démarre une opération d'entrées analogiques bufférisée. Ce VI définit une fréquence de balayage, le nombre de balayages à acquérir et les conditions de déclenchement. Le VI commence ensuite l'acquisition.

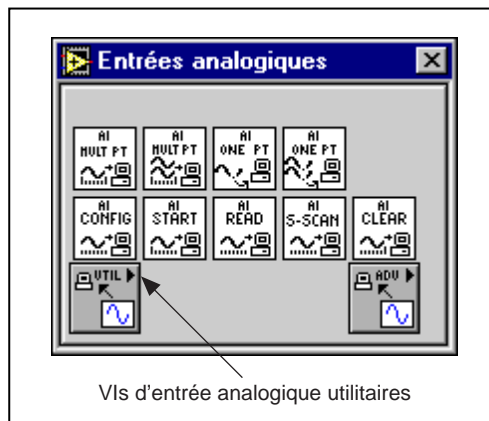


Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour obtenir les gammes, les limites d'entrées et l'ordre du balayage des voies disponibles sur votre périphérique DAQ National Instruments.

VIs Utilitaires d'entrées analogiques

Ce chapitre décrit les VIs Utilitaires d'entrées analogiques. Ces VIs, “AI-Lire un balayage” (AI Read One Scan.vi), “AI-Balayer un signal” (AI Waveform Scan.vi) et “AI-Balayer en continu” (AI Continuous Scan.vi), sont des solutions simples à des problèmes communs d'entrées analogiques. Les VIs Utilitaires d'entrées analogiques sont des VIs de niveau intermédiaire et dépendent des VIs de niveau avancé. Reportez-vous au chapitre 18, *VIs d'entrées analogiques avancés*, pour plus d'informations sur les entrées, les sorties et leur fonctionnement.

Ouvrez la palette **Utilitaires d'entrées analogiques** à partir de la palette **Fonctions»Acquisition de données»Entrées analogiques**. L'icône que vous devez sélectionner pour accéder aux VIs Utilitaires d'entrées analogiques se trouve dans la rangée inférieure de la palette **Entrées analogiques**, comme illustré ci-dessous.



Gestion d'erreurs

LabVIEW simplifie la gestion d'erreurs dans les VIs Utilitaires d'entrées analogiques intermédiaires. Chaque VI de niveau intermédiaire comporte un cluster d'entrée **entrée d'erreur** et un cluster de sortie **sortie d'erreur**. Les clusters sont composés d'un booléen qui indique la présence d'une erreur, le code d'erreur et le nom du VI qui a retourné l'erreur. Si **entrée d'erreur** indique une erreur, le VI transmet les informations d'erreur au cluster **sortie d'erreur** et arrête l'exécution.

Quand vous utilisez n'importe quel VI Utilitaire d'entrées analogiques dans une boucle While, arrêtez la boucle si le booléen **état** du cluster **sortie d'erreur** est Vrai (TRUE). Si vous câblez le cluster d'erreur au VI "Gestionnaire général d'erreurs" (General Error Handler.vi), le VI déchiffre les informations relatives à l'entrée d'erreur et les affiche dans une boîte de dialogue.

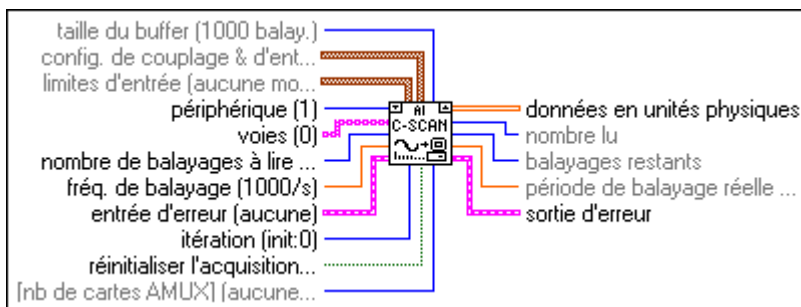
Le VI "Gestionnaire général d'erreurs" se trouve dans la palette **Fonctions»Temps et dialogue** de LabVIEW. Pour plus d'informations sur ce VI, reportez-vous au chapitre 10, *Fonctions de temps, de dialogue et d'erreurs*.

Descriptions des VIs Utilitaires d'entrées analogiques

Les VIs Utilitaires d'entrées analogiques suivants sont disponibles dans la sous-palette **Utilitaires d'entrées analogiques**.

AI-Balayer en continu (AI Continuous Scan.vi)

Effectue des mesures continues et séquencées d'un groupe de voies, stocke les données dans un buffer circulaire et retourne un nombre spécifié de mesures de balayage lors de chaque appel.





terminal
d'itération

Le VI “AI-Balayer en continu” (AI Continuous Scan.vi) balaye un groupe de voies indéfiniment, comme dans le cas d’applications d’enregistrement de données. Placez le VI dans une boucle While et câblez le terminal d’itération de la boucle à l’entrée d’**itération** du VI.

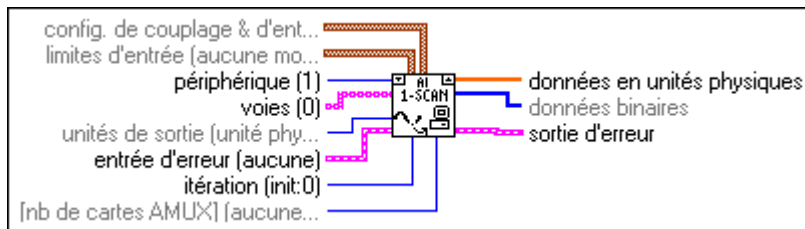
Câblez également la condition qui termine la boucle à l’entrée **réinitialiser l’acquisition**, en inversant le signal si nécessaire pour lire Vrai (TRUE) lors de la dernière itération. Lors de la première itération (0), le VI appelle le VI “AI-Configurer” (AI Config.vi) pour configurer le groupe de voies et le matériel, et alloue un buffer de données ; il appelle ensuite le VI “AI-Démarrer” (AI Start.vi) pour définir le taux de balayage et démarrer l’acquisition. A chaque itération, le VI appelle le VI “AI-Lire” (AI Read.vi) pour récupérer le nombre de mesures spécifié par **nombre de balayages à lire**, effectue leur mise à l’échelle et retourne les données sous la forme d’un tableau de valeurs mises à l’échelle. A la dernière itération (quand **réinitialiser l’acquisition** est Vrai (TRUE) ou si une erreur survient, le VI appelle le VI “AI-Réinitialiser” (AI Clear.vi) pour réinitialiser toute acquisition en cours. Vous n’avez pas besoin d’appeler le VI “AI-Balayer en continu” (AI Continuous Scan.vi) en dehors d’une boucle, mais si vous le faites, vous pouvez laisser les entrées **itération** et **réinitialiser l’acquisition** non câblées.

Lors d’un appel du VI “AI-Balayer en continu” (AI Continuous Scan.vi) dans une boucle pour lire des portions du buffer alors qu’une acquisition est en cours, vous devez lire les données avant qu’elles ne soient écrasées par les données nouvellement acquises. La sortie **balayages restants** indique combien de données le VI a acquises mais n’ont pas encore été lues. Si les balayages restants augmentent vite, vos nouvelles données peuvent éventuellement écraser les anciennes. Récupérez plus souvent les données ou modifiez la **taille du buffer**, le **taux de balayage** ou le **nombre de balayages à lire** pour résoudre ce problème.

Reportez-vous à l’Annexe B, *Capacités matérielles DAQ*, pour obtenir les gammes de voies, les limites d’entrées et l’ordre de balayage disponibles sur votre périphérique DAQ.

AI-Lire un balayage (AI Read One Scan.vi)

Mesure les signaux sur les voies spécifiées et retourne les mesures sous la forme d’un tableau de valeurs binaires ou mises à l’échelle.





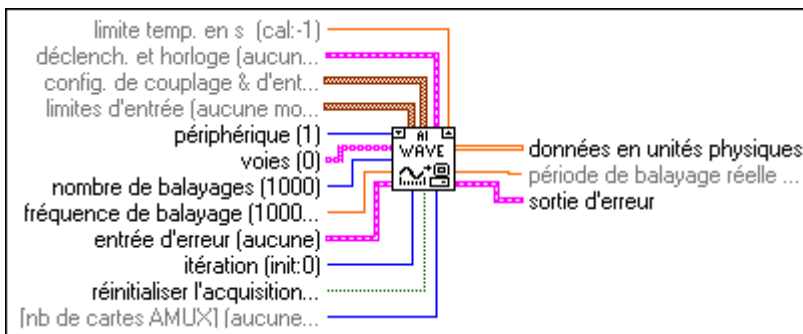
Le VI “AI-Lire un balayage” (AI Read One Scan.vi) effectue une mesure immédiate d’un groupe de plusieurs voies. Si vous placez le VI dans une boucle pour effectuer plusieurs mesures d’un groupe de voies, connectez le terminal d’itération de la boucle au paramètre d’**itération** du VI.

Lors de la première itération (0), le VI appelle le VI “AI-Configurer” (AI Config.vi) pour configurer le groupe de voies et le matériel puis appelle le VI “AI-Balayage simple” (AI Single Scan.vi) pour mesurer et indiquer les résultats. Lors des itérations suivantes, le VI évite des configurations non nécessaires et n’appelle que le VI “AI-Balayage simple” (AI Single Scan.vi). Si vous appelez le VI “AI-Lire un balayage” (AI Read One Scan.vi) une fois pour obtenir une mesure unique d’un groupe de voies, le paramètre d’**itération** peut ne pas être câblé.

Reportez-vous à l’Annexe B, *Capacités matérielles DAQ*, pour obtenir les gammes de voies, les limites d’entrées et l’ordre de balayage disponibles sur votre périphérique DAQ.

AI-Balayer un signal (AI Waveform Scan.vi)

Acquiert le nombre spécifié de balayages à la fréquence de balayage spécifiée et retourne toutes les données acquises. L’acquisition peut être déclenchée.



Le VI “AI-Balayer un signal” (AI Waveform Scan.vi) acquiert un nombre spécifié de balayages d’un groupe de voies à une fréquence de balayage spécifiée. Si vous placez ce VI dans une boucle pour obtenir plusieurs acquisitions dans un même groupe de voies, connectez le terminal d’itération de la boucle à l’entrée d’**itération** du VI.

Câblez également la condition qui termine la boucle à l’entrée **réinitialiser l’acquisition**, en inversant le signal si nécessaire pour lire Vrai (TRUE) lors de la dernière itération. Lors de la première itération 0, le VI appelle le VI “AI-Configurer” (AI Config.vi) pour configurer le groupe de voies et le matériel et allouer un buffer de données. A chaque itération, il appelle le VI “AI-Démarrer” (AI Start.vi) et AI-Lire (AI Read.vi). Le VI “AI-Démarrer” (AI Start.vi) définit la fréquence de balayage, les conditions de déclenchement et démarre l’acquisition. Le VI stocke les mesures dans le buffer à mesure qu’elles sont acquises et le VI “AI-Lire”

(AI Read.vi) récupère les données du buffer, effectue leur mise à l'échelle et retourne toutes les données sous la forme de tableau de valeurs mises à l'échelle. Lors de la dernière itération (quand **réinitialiser l'acquisition** est Vrai (TRUE) ou si une erreur survient, le VI appelle également le VI "AI-Réinitialiser" (AI Clear.vi) pour réinitialiser l'acquisition en cours. Si vous n'appellez le VI "AI-Balayer un signal" (AI Waveform Scan.vi) qu'une seule fois, vous pouvez laisser les entrées **itération** et **réinitialiser l'acquisition** non câblées.

Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour obtenir les gammes, les limites d'entrées et l'ordre de balayage des voies que vous pouvez utiliser sur votre périphérique DAQ National Instruments.



Remarque *Ces VIs utilisent un registre à décalage non initialisé comme mémoire locale pour se souvenir du taskID correspondant au groupe de voies entre chaque appel du VI. Habituellement, vous utilisez un VI à un emplacement du diagramme, cependant, si vous l'utilisez à plusieurs endroits dans un même diagramme, les instances multiples du VI partagent le même taskID. Tous les appels d'un de ces VIs configurent, lisent des données, ou réinitialisent la même acquisition. Si vous souhaitez utiliser chaque VI à plusieurs endroits et que chaque instance correspond à un taskID différent (par exemple, lorsque vous mesurez deux dispositifs en même temps), enregistrez une copie du VI avec le même nom (AI-Balayer un signal R, par exemple) et rendez ré-entrant votre nouveau VI.*

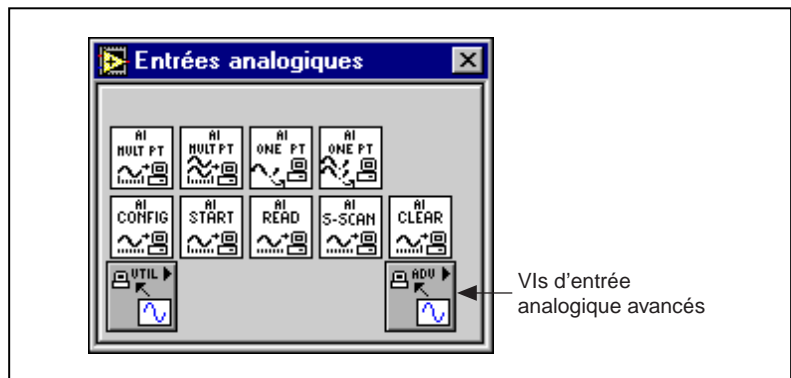


Remarque *Pour tous les VIs Utilitaires d'entrées analogiques, si votre programme effectue une itération plus de $2^{31} - 1$ fois, ne câblez pas l'entrée itération au terminal d'itération de la boucle. Définissez plutôt itération sur 0 pour la première réalisation de la boucle, puis sur n'importe quelle valeur positive pour toutes les autres itérations. Le VI reconfigure et redémarre si l'itération ≤ 0 .*

Vis d'entrées analogiques avancés

Ce chapitre présente des descriptions des VIs d'entrées analogiques avancés. Ces VIs constituent une interface avec le logiciel NI-DAQ et la base des VIs d'entrées analogiques simples, utilitaires et intermédiaires.

Vous pouvez accéder à la palette **Entrées analogiques avancées** en sélectionnant **Fonctions»Acquisition de données»Entrées analogiques avancées**. L'icône que vous devez sélectionner pour accéder aux VIs d'entrées analogiques avancés est dans la rangée du bas de la palette **Entrées analogiques**, comme illustré ci-dessous.

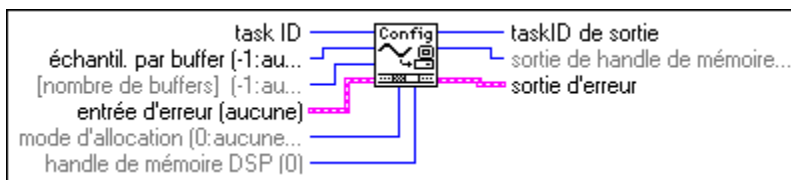


Descriptions des VIs d'entrées analogiques avancés

Les VIs d'entrées analogiques avancés suivants sont disponibles.

AI-Configurer un buffer (AI Buffer Config.vi)

Alloue la mémoire pour que LabVIEW puisse stocker des données acquises en attendant que le VI "AI-lire un buffer" (AI Buffer Read.vi) vienne les lire. LabVIEW se réfère aux buffers alloués par le VI "AI-Configurer un buffer" (AI Buffer Config.vi) en tant que buffers internes car vous ne pouvez pas y accéder directement.



Remarque

Exécuter le VI "AI-Contrôler" (AI Control.vi) avec le code de contrôle défini sur 4 (réinitialiser) équivaut à exécuter le VI "AI-Configurer un buffer" (AI Buffer Config.vi) avec le mode d'attribution défini sur 1. En effet, ces deux VIs désallouent les buffers de données d'entrées analogiques internes. Néanmoins, les acquisitions qui utilisent une carte DSP ou une carte d'extension sont une exception. Le VI "AI-Contrôler" (AI Control.vi) ne désalloue pas la mémoire DSP lors de la réinitialisation de l'acquisition. Vous devez explicitement appeler le VI "AI-Configurer un buffer" (AI Buffer Config.vi) pour désallouer des buffers d'acquisition DSP.

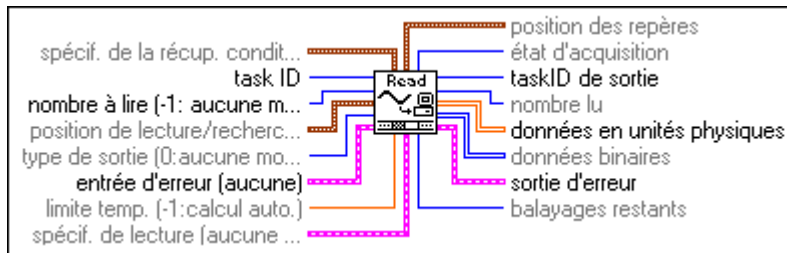
Le tableau 18-1 liste les paramètres et les gammes par défaut pour le VI "AI-Configurer un buffer" (AI Buffer Config.vi). La première rangée présente les valeurs pour la plupart des périphériques et les autres rangées présentent les valeurs pour les périphériques qui sont des exceptions à la règle.

Tableau 18-1. Paramètres et gammes spécifiques au périphérique pour le VI “AI-Configurer un buffer” (AI Buffer Config.vi)

Périphérique	Balayages par buffer		Nombre de buffers		Mode d'attribution	
	Paramètres par défaut	Gamme	Paramètres par défaut	Gamme	Paramètres par défaut	Gamme
La plupart des périphériques	100	$0, n \geq 3$	1	0, 1	2	1, 2
Lab-NB Lab-LC	100	$n \geq 3$	1	0, 1	2	1, 2
NB-A2000 NB-A2100 NB-A2150	100	$n \geq 0$	1	$n \geq 0$	2	1, 2
Périphériques 5102	100	$n \geq 3$	1	1	2	1, 2

AI-Lire un buffer (AI Buffer Read.vi)

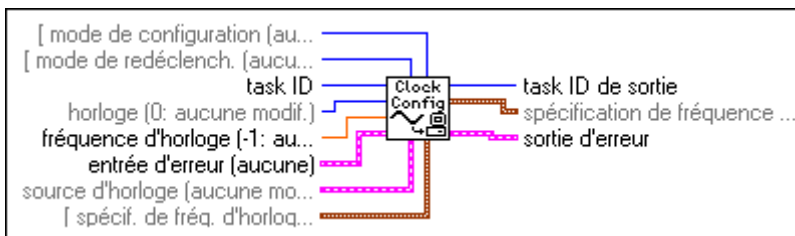
Retourne des données stockées dans le(s) buffer(s) de données internes.



Remarque *Lorsque le VI effectue une lecture à partir du repère de déclenchement, il ne retourne pas de données jusqu'à ce que l'acquisition s'achève pour le buffer comprenant le déclenchement.*

AI-Configurer une horloge (AI Clock Config.vi)

Définit les fréquences d'horloge de voies et de balayages.



Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour obtenir les horloges disponibles avec votre périphérique DAQ.

Pour les périphériques ne disposant que d'une horloge de voies (Lab-LC, Lab-NB, NB-MIO-16, Lab-PC+, PCI-1200, PC-LPM-16, DAQCard-500, DAQCard-700 et DAQCard-1200), vous ne pouvez pas paramétrer les fréquences d'horloge de voies et de balayage. Paramétrer l'une redéfinit l'autre, car l'horloge de voies est égale à l'horloge de balayage/nombre de voies à balayer.

Pour les périphériques ne disposant pas d'horloge de voies (NB-A2000, NB-A2100 et NB-A2150), paramétrer une horloge de voies retourne une erreur.

Si vous spécifiez une valeur égale à 0 pour la fréquence d'horloge de balayage, le balayage à intervalles se désactive et le balayage des voies (ou balayage circulaire) se poursuit avec la fréquence d'horloge de balayage. Cette option n'affecte que les périphériques munis d'horloges de voies et de balayage indépendantes.

La fréquence d'horloge représente la fréquence à laquelle LabVIEW échantillonne les données ou effectue l'acquisition de balayages. Vous pouvez définir la fréquence d'horloge de trois façons : grâce aux paramètres **fréquence d'horloge** ou **période d'horloge**, ou bien par l'intermédiaire des paramètres **source de base de temps**, **signal de base de temps** et **diviseur de base de temps**. Le VI recherche ces paramètres dans cet ordre et définit la fréquence d'horloge à l'aide du premier dont la valeur n'est pas égale à -1.

Le tableau 18-2 liste les paramètres et gammes par défaut pour les commandes du VI “AI-Configurer une horloge” (AI Clock Config.vi).

Tableau 18-2. Paramètres et gammes spécifiques aux périphériques pour le VI “AI-Configurer une horloge” (AI Clock Config.vi)

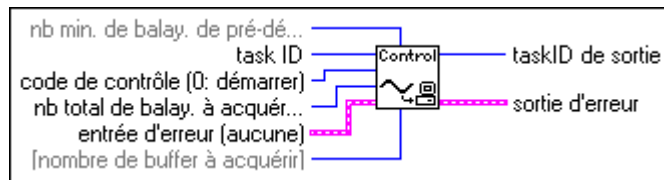
Périphérique	Mode de configuration		Mode de redéclenchement	Type d'horloge		Source d'horloge	
	Paramètres par défaut	Gamme	Paramètres par défaut	Paramètres par défaut	Gamme	Paramètres par défaut	Gamme
AT-MIO-16E1 AT-MIO-16E-2 AT-MIO-64E-1 NEC-MIO-16E-4 PCI-MIO-16E-1 PCI-MIO-16E-4 PCI-MIO-16XE-10	1	1, 3	non supporté	1	1, 2	1	1, 2 $4 \leq n \leq 11$
PCI-6110E PCI-6111E	1	1, 3	non supporté	1	1	1	1, 2 $4 \leq n \leq 11$
AT-MIO-16E-10 AT-MIO-16DE-10 AT-MIO-16XE-50 PCI-MIO-16XE-50	1	1, 3	non supporté	1	1, 2	1	1, 2 $4 \leq n \leq 9$
NB-A2150 NB-A2100 NB-A2000	1	1, 3	non supporté	1	1	1	$1 \leq n \leq 3$
Périphériques DSA	1	1, 3	non supporté	1	1	1	1
PC-LPM-16 DAQCard-500 DAQCard-516 DAQCard-700 Lab-PC	1	1, 3	non supporté	1	1, 2	1	1, 2
Lab-LC Lab-NB NB-MIO-16	1	1, 3	non supporté	1	2	1	1, 2

Tableau 18-2. Paramètres et gammes spécifiques aux périphériques pour le VI “AI-Configurer une horloge” (AI Clock Config.vi) (Suite)

Périphérique	Mode de configuration		Mode de redéclenchement	Type d'horloge		Source d'horloge	
	Paramètres par défaut	Gamme	Paramètres par défaut	Paramètres par défaut	Gamme	Paramètres par défaut	Gamme
5102	1	1, 3	non supporté	1	1	1	1, 6
5911, 5912	1	1, 3	non supporté	1	1,2	1	$1 \leq n \leq 3$
Tous les autres périphériques	1	1, 3	non supporté	1	1, 2	1	$1 \leq n \leq 3$


AI-Contrôler (AI Control.vi)

Contrôle les tâches d'entrées analogiques et spécifie la quantité de données à acquérir.



Remarque

Vous ne pouvez pas utiliser ce VI pour commencer une acquisition lorsque vous utilisez une carte PC-LPM-16, DAQCard-500, DAQCard-700 ou des séries Lab et 1200 pour balayer plusieurs voies SCXI en mode multiplexé. Ces instances spéciales exigent le VI “AI-BalayageUnique” (AI SingleScan.vi) pour acquérir des données. (Pour plus d’informations sur le VI “AI-BalayageUnique” (AI SingleScan.vi), reportez-vous à sa description dans ce chapitre.) Néanmoins, vous pouvez utiliser le VI “AI-Contrôler” (AI Control.vi) pour une carte PC-LPM-16, DAQCard-500, DAQCard-700 ou des séries Lab et 1200 lorsque vous balayez des voies SCXI en mode parallèle ou que vous échantillonnez une seule voie SCXI en mode multiplexé. Vous pouvez utiliser ce VI si vous disposez d’un périphérique MIO qui balaye des voies SCXI dans l’un des deux modes.

 **Remarque** Les acquisitions non bufférisées ne sont pas supportées par les périphériques suivants.

- (Macintosh) NB-A2000
- (Macintosh) NB-A2100
- (Macintosh) NB-A2150

Le tableau 18-3 liste les paramètres et gammes par défaut pour le VI "AI-Contrôler" (AI Control.vi).

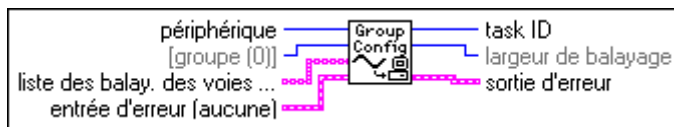
Tableau 18-3. Paramètres et gammes spécifiques aux périphériques pour le VI "AI-Contrôler" (AI Control.vi)

Périphérique	Code de contrôle		Balayage total à acquérir		Nombre minimum de balayage à acquérir avant le déclenchement		Nombre de buffers à acquérir	
	DS*	R*	DS*	R*	DS*	R*	DS*	R*
NB-A2000 NB-A2150	0	0, 1, 4	0	0, n≥0	0	0, n≥3	1	n≥0
PC-LPM-16 DAQCard-500 DAQCard-700	0	0, 1, 4	0	0, n≥3	0	non supporté	1	1
MIO-Série E	0	0, 1, 4	0	0, n≥3	0	0, n≥3	1	1
Périphériques 5102	0	0, 1, 4	0	n≥0	0	n≥0	1	1
5911, 5912	0	0,4	0	n≥1	0	n≥0	1	1
Tous les autres périphériques	0	0, 1, 4	0	0, n≥3	0	n≥0	1	1

* DS = Paramètre par défaut ; R = Gamme

AI-Configurer un groupe (AI Group Config.vi)

Définit les voies d'un groupe et les alloue.



Reportez-vous à l'Annexe B, Capacités matérielles DAQ, pour obtenir les gammes et l'ordre de balayage des voies avec votre périphérique DAQ.

Le tableau 18-4 liste les paramètres et gammes par défaut pour le VI "AI-Configurer un groupe" (AI Group Config.vi). La première rangée du tableau présente les valeurs pour la plupart des périphériques et les autres rangées présentent les valeurs pour les périphériques qui sont des exceptions à la règle.

Tableau 18-4. Paramètres et gammes spécifiques aux périphériques pour le VI "AI-Configurer un groupe" (AI Group Config.vi)

Périphérique	Groupe		Liste des balayages de voies	
	Paramètre par défaut	Gamme	Paramètre par défaut	Gamme
La plupart des périphériques Windows	0	$0 \leq n \leq 15$	toutes les voies	$0 \leq n \leq 15$
La plupart des périphériques Macintosh	0	$0 \leq n \leq 15$	toutes les voies	$0 \leq n \leq 15$
AT-MIO-64F-5 AT-MIO-64E-1*	0	$0 \leq n \leq 15$	toutes les voies	$0 \leq n \leq 63$
Lab-PC+, PCI-1200, DAQCard-1200	0	$0 \leq n \leq 15$	toutes les voies	$0 \leq n \leq 7$
Lab-LC, Lab-NB	0	$0 \leq n \leq 15$	toutes les voies	$0 \leq n \leq 7$
NB-A2000, NB-A2150	0	$0 \leq n \leq 15$	toutes les voies	$0 \leq n \leq 3$
NB-A2100	0	$0 \leq n \leq 15$	toutes les voies	0, 1
Périphériques 5102	0	$0 \leq n \leq 15$	toutes les voies	0, 1
PCI-4452, PCI-4451	0	$0 \leq n \leq 15$	toutes les voies	0, 1
PCI-4452, PCI-4552	0	$0 \leq n \leq 15$	toutes les voies	$0 \leq n \leq 3$

* Les voies valides pour l'AT-MIO-64E-1 en mode différentiel sont 0-7, 16-23, 34-39, 48-55.



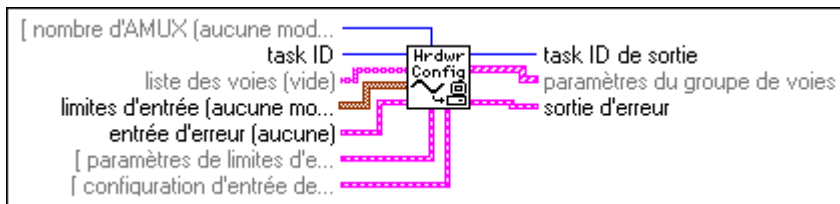
Remarque *Les périphériques Lab-LC, Lab-NB, Lab-PC+, PCI-1200, PC-LPM-16, DAQCard-500, DAQCard-700 et DAQCard-1200 doivent balayer des listes de voies qui comportent plusieurs voies à partir de la voie n ($n \geq 0$) jusqu'à la voie 0 en ordre séquentiel, en incluant toutes les voies comprises entre n et 0. Les cartes NB-A2000, NB-A2150, EISA-A2000 et AT-A2150 n'acceptent que la liste des balayages suivante : (0), (1), (2), (3), (0, 1), (2, 3) et (0, 1, 2, 3). Le périphérique NB-A2100 accepte la liste des balayages suivante : (0), (1), (0, 1) et (1, 0).*

La gamme liste des balayages de voies figurant ci-dessus n'est valable que pour le mode asymétrique. Reportez-vous à l'Annexe B, Capacités matérielles DAQ, pour obtenir les gammes valides pour les voies en mode différentiel.

Les modules SCXI en mode multiplexé doivent balayer les voies dans l'ordre ascendant, en commençant par n'importe quelle voie du module. L'ordre de module que vous spécifiez peut être arbitraire. Les modules SCXI en mode parallèle doivent suivre les restrictions du périphérique DAQ relatives à l'ordre de la liste des voies. Reportez-vous à la section *Adressage de voies, de portes et de compteurs* au chapitre 3, *Concepts de base de l'acquisition de données LabVIEW*, dans le *Manuel de base d'acquisition de données LabVIEW* pour des informations sur la syntaxe de la chaîne de caractères adressant une voie SCXI.

AI-Configurer un matériel (AI Hardware Config.vi)

Configure soit les limites supérieure et inférieure, soit la gamme, la polarité et le gain. Le VI "AI-Configurer un matériel" (AI Hardware Config.vi) configure également le couplage, le mode d'entrée et le nombre de périphériques AMUX-64T. L'utilitaire de configuration détermine les paramètres par défaut pour les paramètres de ce VI.



Vous pouvez utiliser ce VI pour récupérer les paramètres courants en ne câblant que **taskID** ou en câblant **taskID** et **liste de voies**. Si **liste de voies** est vide, le VI configure les voies *par groupe*, ce qui veut dire que la configuration s'applique à toutes les voies du groupe. Quand vous spécifiez une ou plusieurs voies dans **liste de voies**, le VI configure les voies *par voie*. Ceci veut dire que la configuration ne s'applique qu'aux voies que vous spécifiez. Ce VI retourne toujours les paramètres courants pour tout le groupe.

Dans le cas d'une configuration par voie, la **liste de voies** peut comprendre une voie ou plus. Les voies de la **liste de voies** doivent appartenir au groupe nommé par **taskID**. Vous spécifiez

les voies de la même façon que vous les spécifiez pour le VI “AI-Configurer un groupe” (AI Group Config.vi). Si vous faites l'acquisition de plusieurs échantillons d'une voie dans un balayage et que vous souhaitez changer la configuration du matériel sur cette voie pour chaque échantillon, vous devez fournir les paramètres pour chaque instance de la voie dans le balayage. Si un élément de la **liste de voies** spécifie plusieurs voies, l'élément correspondant des autres tableaux s'applique à toutes ces voies.

Le VI applique les valeurs comprises dans les tableaux de configuration (**limites d'entrée supérieures, limites d'entrée inférieures, couplage, gamme, polarité, gain et mode**) aux voies du groupe (si vous avez une configuration par groupe) ou aux voies de la **liste de voies** (si vous avez une configuration par voie) de la façon suivante. Le VI applique les valeurs listées en premier dans les tableaux (à l'indice 0) à la première voie de groupe ou des voies listées à l'indice 0 de **liste de voies**. Le VI applique les valeurs listées en second dans le tableau de configuration (à l'indice 1) à la seconde voie du groupe ou des voies listées à l'indice 1 de **liste de voies**. Le VI continue d'appliquer les valeurs de cette manière jusqu'à ce que les tableaux soient vides. Si les voies du groupe ou **liste de voies** ne sont pas configurées, le VI applique les dernières valeurs des tableaux à toutes les voies non configurées.

Le tableau 18-5 présente des exemples de cette méthode. Le paramètre **liste de balayages de voies**, qui fait partie du VI “AI-Configurer un groupe” (AI Group Config.vi), est utilisé dans le tableau suivant.

Tableau 18-5. Configuration des voies pour “AI-Configurer un matériel” (AI Hardware Config.vi)

Type de configuration	Valeurs de tableau	Résultats
Groupe	<p>Groupe de la liste de balayages de voies = 1, 3, 4, 5, 7</p> <p>liste de voies est vide</p> <p>limite d'entrée inférieure [0] = -1,0</p> <p>limite d'entrée supérieure [0] = +1,0</p>	Toutes les voies du groupe ont des limites d'entrée de -1,0 à +1,0.
Groupe	<p>Groupe de la liste de balayages de voies = 1, 3, 4, 5, 7</p> <p>liste de voies est vide</p> <p>limite d'entrée inférieure [0] = -1,0</p> <p>limite d'entrée supérieure [0] = +1,0</p> <p>limite d'entrée inférieure [1] = 0,0</p> <p>limite d'entrée supérieure [1] = +5,0</p> <p>limite d'entrée inférieure [2] = -10,0</p> <p>limite d'entrée supérieure [2] = +10,0</p>	La voie 1 a des limites d'entrée de -1,0 à +1,0. La voie 3 a des limites d'entrée de 0,0 à +5,0. Les voies 4, 5 et 7 ont des limites d'entrée de -10,0 à +10,0.

Tableau 18-5. Configuration des voies pour "AI-Configurer un matériel" (AI Hardware Config.vi) (Suite)

Type de configuration	Valeurs de tableau	Résultats
Voie	Groupe de la liste de balayages de voies = 1, 3, 4, 5, 7 liste de voies [0] = 1 liste de voies [1] = 3:5 limite d'entrée inférieure [0] = -1,0 limite d'entrée supérieure [0] = +1,0	Les voies 1, 3, 4 et 5 ont des limites d'entrée de -1,0 à +1,0. La voie 7 a des limites d'entrée par défaut définies par l'utilitaire de configuration. Elle n'est pas modifiée car elle ne figure pas dans la liste de voies .
Voie	Groupe de la liste de balayages de voies = 1, 3, 4, 5, 7 liste de voies [0] = 1 liste de voies [1] = 3:5 limite d'entrée inférieure [0] = -1,0 limite d'entrée supérieure [0] = +1,0 limite d'entrée inférieure [1] = 0,0 limite d'entrée supérieure [1] = +5,0	La voie 1 comporte des limites d'entrée de -1,0 à +1,0. Les voies 3, 4 et 5 ont des limites d'entrée de 0,0 à +5,0. La voie 7 comporte des limites d'entrée par défaut définies par l'utilitaire de configuration.
Groupe	Groupe de la liste de balayages de voies = 0, 1, 0, 1 liste de voies est vide limite d'entrée inférieure [0] = -1,0 limite d'entrée supérieure [0] = +1,0 limite d'entrée inférieure [1] = -1,0 limite d'entrée supérieure [1] = +1,0 limite d'entrée inférieure [2] = -10,0 limite d'entrée supérieure [2] = +10,0 limite d'entrée inférieure [3] = -10,0 limite d'entrée supérieure [3] = +10,0	Les voies 0 et 1 comportent des limites d'entrée de -1,0 à +1,0 la première fois qu'elles sont échantillonnées et des limites d'entrée de -10,0 à +10,0 la seconde fois qu'elles sont échantillonnées.

La **gamme**, la **polarité** et le **gain** déterminent les limites d'entrée supérieures et inférieures. Lorsque vous câblez des tableaux de **limites d'entrée** valides (c'est-à-dire, des tableaux dont la longueur est supérieure à zéro) le VI choisit des gammes d'entrée, des polarités et des gains appropriés pour obtenir ces **limites d'entrée**. Le VI ignore les tableaux de **gamme**, **polarité** et **gain**.

Si vous ne câblez pas les tableaux de **limites d'entrée**, le VI vérifie la **gamme**, la **polarité** et le **gain**. Quand le VI trouve un tableau, il définit la propriété d'entrée correspondante aux valeurs dans le tableau. Quand le VI ne trouve pas de tableau, il ne modifie pas la propriété d'entrée correspondante.

Pour quelques périphériques et modules SCXI, des cavaliers intégrés permettent de définir la **gamme**, la **polarité** et/ou le **gain**. LabVIEW ne change pas les paramètres réglés par cavalier lorsque vous spécifiez les **limites d'entrée**. Si LabVIEW ne peut pas obtenir les **limites d'entrée** souhaitées à l'aide des paramètres réglés par cavaliers, il retourne une mise en garde.

Pour ne pas tenir compte des valeurs définies par les positions courantes des cavaliers, vous devez appeler le VI "AI-Configurer un matériel" (AI Hardware Config.vi) et spécifier la **gamme**, la **polarité**, et/ou le **gain** explicitement. L'utilitaire de configuration détermine le réglage initial pour ces paramètres (la valeur par défaut est le paramètre du cavalier réglé par le fabricant).

Si une paire de valeurs de **limites d'entrée** est égale à 0, le VI ne change pas les **limites d'entrée**.

Les configurations matérielles de la voie SCXI sont en fait une combinaison des paramètres du module SCXI et de ceux du périphérique DAQ. Elles exigent des considérations spéciales. La manière dont vous spécifiez les voies indique si LabVIEW doit modifier les paramètres du module SCXI et/ou les paramètres du périphérique DAQ. Le paramètre des **limites d'entrée** s'applique toujours au chemin d'acquisition en entier.

Quand vous effectuez une configuration par groupe, LabVIEW peut modifier les paramètres du module SCXI et du périphérique DAQ. Dans ce cas, le **gain** s'applique au chemin entier et est le produit du gain de la voie SCXI et du gain de la voie du périphérique d'acquisition. LabVIEW définit le gain maximal nécessaire au module SCXI et ajoute ensuite le gain du périphérique DAQ si nécessaire.

Quand vous effectuez une configuration par voie, vous pouvez spécifier les voies de trois façons différentes. La première est de spécifier le chemin entier :

```
OB0!SC1!MD1!CH0:7
```

Vous pouvez également spécifier le chemin à l'aide de noms de voies configurées par l'Assistant Entrées/Sorties DAQ, comme l'illustre l'exemple ci-dessous.

```
temperature
```

Si vous utilisez l'une de ces deux méthodes, LabVIEW peut modifier les paramètres des périphériques SCXI et DAQ et le **gain** s'applique au produit créé par le gain de la voie SCXI et le gain de la voie du périphérique DAQ. LabVIEW définit le gain maximal nécessaire au module SCXI et ajoute ensuite le gain du périphérique DAQ si nécessaire.

La seconde méthode est de ne spécifier que la voie SCXI, comme l'illustre l'exemple ci-dessous.

```
SC1!MD1!CH0:7
```

Cette spécification indique que LabVIEW ne doit modifier que les paramètres du SCXI. Le **gain** ne s'applique donc qu'à la voie SCXI.

La troisième méthode est de ne spécifier que la voie du périphérique d'acquisition, comme l'illustre l'exemple ci-dessous.

OB0

Dans ce cas, LabVIEW ne modifie que les paramètres du périphérique DAQ. Le paramètre **gain** ne s'applique qu'à la voie de la carte d'acquisition.

Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour obtenir les gammes de voies, les limites d'entrée et l'ordre de balayage disponibles sur votre périphérique DAQ.

Les tableaux 18-6 à 18-9 listent les paramètres et gammes par défaut pour le VI "AI-Configurer un matériel" (AI Hardware Config.vi). Un tilde (~) indique que le paramètre est configurable par groupe seulement. Vous ne pouvez donc pas configurer par voie. La première rangée de ces tableaux présente les valeurs pour la plupart des périphériques, les autres rangées présentent les valeurs pour les périphériques qui sont des exceptions à la règle. Si vous n'avez pas défini les paramètres par défaut avec l'utilitaire de configuration, utilisez les paramètres par défaut figurant dans ces tableaux.


Tableau 18-6. Paramètres et gammes spécifiques au périphérique pour le VI "AI-Configurer un matériel" (AI Hardware Config.vi)

Périphérique	Cluster de configuration de l'entrée de voie				Nombre d'AMUX		Liste de voies
	Couplage		Mode d'entrée ~				
	DS*	R*	DS*	R*	DS*	R*	DS*
La plupart des périphériques	1	1	1	$1 \leq n \leq 3$	0	$0 \leq n \leq 4$	vide
NB-A2000	2	1, 2	2	2	0	0	vide
PC-LPM-16, Lab-LC, Lab-NB	1	1	2	2	0	0	vide
Périphériques Séries 1200 et Lab	1	1	2	$1 \leq n \leq 3$	0	0	vide
AT-MIO-16X, AT-MIO-64F-5	1	1	1 (no ~)	$1 \leq n \leq 3$	0	$0 \leq n \leq 4$	vide

Tableau 18-6. Paramètres et gammes spécifiques au périphérique pour le VI "AI-Configurer un matériel" (AI Hardware Config.vi) (Suite)

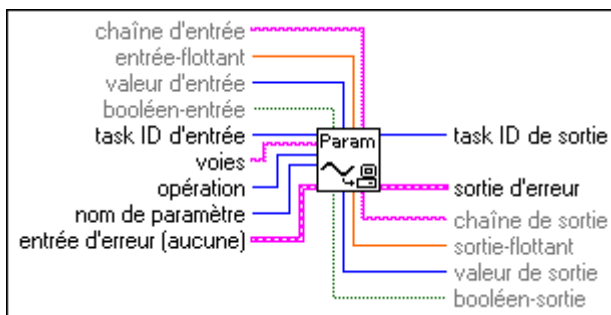
Périphérique	Cluster de configuration de l'entrée de voie				Nombre d'AMUX		Liste de voies
	Couplage		Mode d'entrée ~				
	DS*	R*	DS*	R*	DS*	R*	DS*
NB-A2100, NB-A2150	1	1, 2	2	2	0	0	vide
DAQCard-500, DAQCard-516, DAQCard-700	1	1	2	1, 2	0	0	vide
Périphériques 5102	1	1,2	2	2	0	0	vide
PCI-6110E, PCI-6111E, PCI-4451, PCI-4551, PCI-4452, PCI-4552	1	1, 2	1	1	0	0	vide

* DS = Paramètre par défaut ; R = Gamme

 **Remarque** Les voies 0 et 1 et les voies 2 et 3 doivent avoir le même couplage pour le NB-A2150.

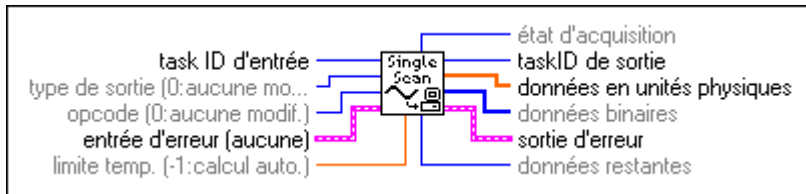
AI-Paramètre (AI Parameter.vi)

Configure et récupère divers paramètres associés aux entrées analogiques d'une opération réalisée par un périphérique qui ne sont pas traitées par les autres VIs d'entrées analogiques.



AI-BalayageUnique (AI SingleScan.vi)

Retourne un balayage de données. Si vous démarrez une acquisition avec le VI “AI-Contrôler” (AI Control.vi), ce VI lit un balayage des données stockées dans le buffer interne. Sous Macintosh et Windows, le VI lit dans la FIFO intégrée dans le cas d’une acquisition non bufférisée. Si vous n’avez pas démarré une acquisition, ce VI commence l’acquisition, récupère un balayage et termine l’acquisition. La configuration de groupe détermine les voies que le VI échantillonne. Ce VI ne supporte pas les périphériques 5102, DSA et 59xx.



Si vous n’appellez pas le VI “AI-Contrôler” (AI Control.vi), ce VI commence un balayage unique en utilisant la fréquence d’horloge de voie la plus rapide et la plus sûre. Vous pouvez néanmoins modifier cette fréquence d’horloge à l’aide du VI “AI-Configurer une horloge” (AI Clock Config.vi).

Si vous exécutez le VI “AI-Contrôler” (AI Control.vi) avec **code de contrôle** défini sur 0 (Démarrer), un signal d’horloge démarre les balayages.

Si vous souhaitez des conversions séquencées par une horloge, vous devez utiliser le VI “AI-Configurer une horloge” (AI Clock Config.vi) et définir la source d’horloge en externe.

Si les sources d’horloge sont internes et que vous n’allouez pas de mémoire, une acquisition séquencée non bufférisée commence lorsque vous exécutez le VI “AI-Contrôler” (AI Control.vi) avec **code de contrôle** défini à 0. Ce type d’acquisition est utile pour synchroniser des entrées et des sorties analogiques dans une application de contrôle de point par point. Les périphériques suivants ne supportent pas les acquisitions séquencées non bufférisées :

- **(Macintosh)** Lab-NB, Lab-LC, NB-A2000, NB-A2100 et NB-A2150



Remarque *Si un débordement de la FIFO se produit lors d’une acquisition séquencée non bufférisée, LabVIEW redémarre le périphérique.*

Le tableau 18-7 liste les paramètres et les gammes par défaut pour le VI "AI-BalayageUnique" (AI Single Scan.vi).

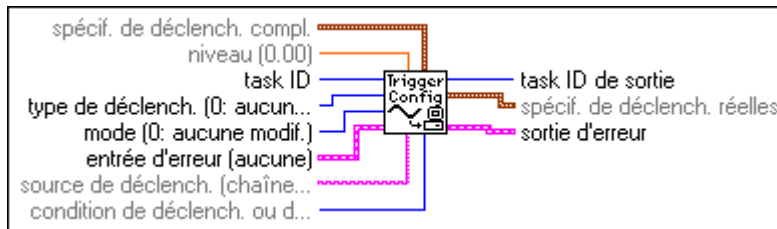
Tableau 18-7. Paramètres et gammes spécifiques aux périphériques pour le VI "AI-BalayageUnique" (AI SingleScan.vi)

Périphérique	Type de sortie		Opcode		Limite temporelle	
	DS	R	DS	R	DS	R
NB-A2000, NB-A2100, NB-A2150	1	$1 \leq n \leq 3$	1	1	variable	$n \geq 0$
Tous les autres périphériques	1	$1 \leq n \leq 3$	1	$1 \leq n \leq 4$	$1 \leq n \leq 4$	$n \geq 0$

* DS = Paramètre par défaut ; R = Gamme

AI-Configurer un déclenchement (AI Trigger Config.vi)

Configure les conditions de déclenchement pour démarrer les horloges de balayage et de voies ainsi que le compteur de balayage.



Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour obtenir des informations sur les types de déclenchements disponibles sur votre périphérique DAQ. Reportez-vous au manuel d'utilisation du périphérique Série E pour obtenir une description détaillée de ses capacités de déclenchement.

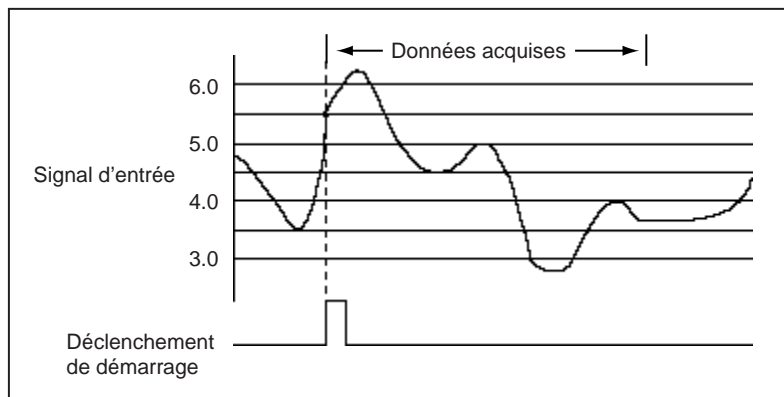
La section suivante présente une description détaillée des types de déclenchement 1 (déclenchement analogique), 2 (déclenchement numérique A) et 3 (déclenchement numérique B) s'appliquant aux trois types d'applications : post-déclenchement, pré-déclenchement avec un démarrage logiciel et pré-déclenchement avec un démarrage matériel. Les autres types de déclenchements sont présentés à la fin de cette section.

Type d'application 1 : acquisition post-déclenchée (déclenchement de démarrage uniquement)

Si **nb total de balay. à acquérir** est ≥ 0 et **balayages de pré-déclench. à acquérir** est égal à 0, vous effectuez une acquisition post-déclenchée. Un **type de déclenchement** égal à

1 ou 2 (respectivement déclenchement analogique ou déclenchement numérique A) démarre l'acquisition (le déclenchement numérique B est non autorisé). Vous devez fournir un déclenchement de démarrage. Reportez-vous au tableau 18-10, sections 2 et 3, pour déterminer la broche par défaut sur laquelle connecter votre signal de déclenchement. Sur certains périphériques, vous pouvez spécifier une autre source grâce au paramètre **source de déclenchement**.

Avec les périphériques de Série E, si vous utilisez un déclenchement analogique et que le signal analogique est connecté à l'une des voies d'entrées analogiques, cette voie doit être la première dans la liste de balayage. Cette restriction n'a pas lieu si vous connectez le signal analogique à PFI0.

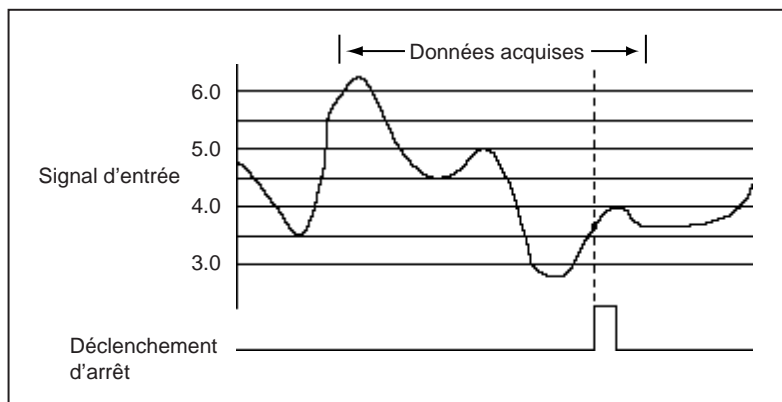


Dans l'illustration ci-dessus, **nb total de balay. à acquérir** est égal à 1000 et **balayages de pré-déclench. à acquérir** est égal à 0. Le déclenchement de démarrage peut provenir du déclenchement numérique A ou d'un déclenchement analogique (**condition de déclenchement ou de pause** = 1: déclenchement sur une pente ou un front montant, **niveau** = 5,5 et **taille de la fenêtre** = 0,2).

Type d'application 2 : acquisition pré-déclenchée (pour tout type de déclenchement)

Si **nb total de balay. à acquérir** et **balayages de pré-déclench. à acquérir** sont tous deux > 0, un **type de déclenchement** égal à 1 ou 2 (respectivement déclenchement analogique ou déclenchement numérique A) démarre l'acquisition de données post-déclenchée après que les données de pré-déclenchement ont été acquises. Le déclenchement est nommé *déclenchement d'arrêt* car l'acquisition continue jusqu'à ce qu'un déclenchement se produise. Un strobe logiciel démarre l'acquisition. Il s'agit donc d'une acquisition avec pré-déclenchement démarrée par logiciel. Vous devez fournir le déclenchement d'arrêt. Reportez-vous au tableau 18-10, sections 2 et 3, pour déterminer la broche par défaut sur laquelle connecter

votre signal de déclenchement. Sur certains périphériques, vous pouvez spécifier une autre source grâce au paramètre **source de déclenchement**.



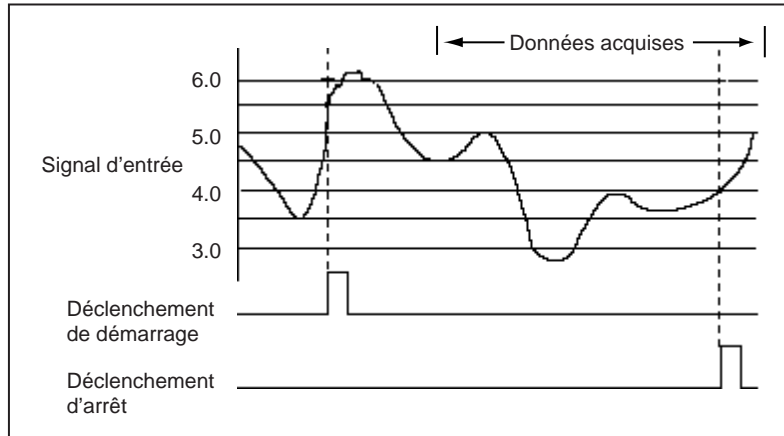
Dans l'illustration précédente, **nb total de balay.** à acquérir est égal à 1000 et **balayages de pré-déclench.** à acquérir est égal à 900. Le déclenchement d'arrêt peut provenir d'un déclenchement numérique A ou d'un déclenchement analogique (**condition de déclenchement ou de pause = 1**: déclenchement sur pente ou front montant, **niveau = 3, 7**, **taille de la fenêtre = 0, 5**).

Avec les périphériques de Série E, si vous utilisez un déclenchement analogique et que le signal analogique est connecté à l'une des voies d'entrées analogiques, cette voie doit être la première dans la liste de balayage (le balayage de voies multiples n'est pas autorisé). Cette restriction n'a pas lieu si vous connectez le signal analogique à PFIO.

Type d'application 3 : acquisition pré-déclenchée (déclenchement de démarrage et d'arrêt)

Le type d'application 3 est rarement utilisé. A moins que vous n'envisagiez de fournir un déclenchement de démarrage et un déclenchement d'arrêt, sautez cette section.

Sur les périphériques MIO, vous pouvez activer le déclenchement de démarrage et le déclenchement d'arrêt. (Vous devez pour cela appeler deux fois le VI "AI-Configurer un déclenchement" (AI Trigger Config.vi). Dans ce cas, un signal de déclenchement analogique ou numérique démarre l'acquisition au lieu d'un strobe logiciel. Il s'agit d'une acquisition pré-déclenchée de démarrage matériel. Vous devez fournir à la fois le déclenchement de démarrage (comme décrit par la section Type d'application 1) et le déclenchement d'arrêt (comme décrit par la section Type d'application 2). Reportez-vous aux tableaux 18-11 et 18-12, sections 2 et 3, pour déterminer la broche par défaut sur laquelle vous connectez votre signal de déclenchement. Sur certains périphériques, vous pouvez spécifier une autre source grâce au paramètre **source de déclenchement**.



Dans l'illustration ci-dessus, **nb total de balay. à acquérir** est égal à 1000 et **balayages de pré-déclench. à acquérir** est égal à 900. Le déclenchement de démarrage peut provenir d'un déclenchement numérique B ou d'un déclenchement analogique (**condition de déclenchement ou de pause** = 1 : déclenchement sur pente ou front montant, **niveau** = 5, 5, **taille de la fenêtre** = 0, 2). Le déclenchement d'arrêt peut provenir d'un déclenchement numérique A ou d'un déclenchement analogique (**condition de déclenchement ou de pause** = 1 : déclenchement sur pente ou front montant, **niveau** = 4, 0, **taille de la fenêtre** = 0, 2). Remarquez que certaines données acquises après le déclenchement de démarrage sont détruites, car les 900 balayages de pré-déclenchement ont tous été réunis et le déclenchement d'arrêt se trouve à plus de 900 balayages après le déclenchement de démarrage.

Quand vous utilisez un déclenchement analogique sur les périphériques de Série E, plusieurs restrictions s'appliquent, comme l'indique le tableau 18-8.

Tableau 18-8. Restrictions pour déclenchement analogique sur les périphériques de Série E

Déclenche-ment de démarrage	Déclenche-ment d'arrêt	Restrictions
A numérique	B numérique	Aucune
B numérique	Analogique	Le signal analogique doit être connecté à PFI0, sauf si vous effectuez le balayage d'une seule voie. Dans ce cas, vous pouvez utiliser l'entrée vers cette voie.
Analogique	A numérique	Le signal analogique doit être le premier dans la liste de balayages s'il est connecté à une voie d'entrées analogiques.

Un **type de déclenchement** égal à 4 (fenêtrage d'horloge de balayage numérique) permet au signal externe TTL de fenêtrer l'horloge de balayage, en arrêtant et en reprenant une acquisition.

L'horloge de voie et l'horloge de balayage sont les mêmes dans la carte NB-MIO-16. Ainsi, si le fenêtrage d'horloge de balayage devient Faux (FALSE), l'horloge de balayage cesse l'opération laissant le balayage inachevé. Quand le fenêtrage d'horloge de balayage devient Vrai (TRUE), l'horloge de balayage recommence immédiatement l'opération, là où elle s'était interrompue. Vous devez câbler votre signal sur la broche EXTGATE.

Un **type de déclenchement** égal à 5 (fenêtrage d'horloge de balayage analogique) permet au signal externe analogique de fenêtrer l'horloge de balayage, en arrêtant et en reprenant une acquisition. Un type de déclenchement égal à 6 vous permet d'utiliser la sortie du circuit de déclenchement analogique (ATCOUT) comme un signal d'usage général. Par exemple, utilisez ATCOUT pour commencer une opération de sortie analogique, ou comptez le nombre de déclenchements analogiques dans ATCOUT.



Remarque *Les types de déclenchements 1, 5 et 6 sur les cartes de Série E emploient le même circuit de déclenchement analogique. Ces trois types peuvent s'activer en même temps, mais le dernier qui est activé dicte le comportement du circuit de déclenchement analogique. Les restrictions des cartes de Série E décrites dans les applications de déclenchement s'appliquent aux trois types de déclenchement.*

Le type de déclenchement 5 sur les cartes de Série E utilise le fenêtrage d'horloge de balayage numérique et le circuit de déclenchement analogique. Ainsi, l'activation de ce type de déclenchement 5 écrase tout autre réglage effectué pour le type de déclenchement 4.

Quelques dispositifs supportent le déclenchement numérique, mais la source est alors prédéterminée. Le paramètre **source de déclenchement** n'est donc pas valable. Le tableau 18-9 présente les noms des broches du connecteur E/S sur lesquelles vous devriez connecter votre signal de déclenchement numérique.

Tableau 18-9. Sources de déclenchement numérique pour des périphériques pourvus de sources de déclenchement numérique fixes

Périphérique	Post-déclenchement	Pré-déclenchement	
	Broche de déclenchement de démarrage	Broche de déclenchement de démarrage	Broche de déclenchement d'arrêt
MIO-16L/H, MIO-16DL/DH	STARTTRIG*	STARTTRIG*	STOPTRIG
NB-MIO-16L/H	STARTTRIG*	non supporté	non supporté

Tableau 18-9. Sources de déclenchement numérique pour des périphériques pourvus de sources de déclenchement numérique fixes (Suite)

Périphérique	Post-déclenchement	Pré-déclenchement	
	Broche de déclenchement de démarrage	Broche de déclenchement de démarrage	Broche de déclenchement d'arrêt
AT-MIO-16X, AT-MIO-16F-5, AT-MIO-64F-5	EXTTRIG*	EXTTRIG*	EXTTRIG*
Périphériques Séries 1200 et Lab	EXTTRIG	non supporté	EXTTRIG
PC-LPM-16, DAQCard-500, DAQCard-700	non supporté	non supporté	non supporté
NB-A2000, NB-A2100, NB-A2150	EXTTRIG*	non supporté	EXTTRIG*
DSA 45xx	EXTTRIG*	EXTTRIG*	EXTTRIG*

* Sur les AT-MIO-16X, AT-MIO-16F-5 et AT-MIO-64F-5, la même broche sert pour le déclenchement de démarrage et le déclenchement d'arrêt. Reportez-vous au manuel d'utilisation de votre carte pour plus de détails.

Le tableau 18-10 liste les paramètres et gammes par défaut et pour le VI "AI-Configurer un déclenchement" (AI Trigger Config.vi). La première rangée de chaque tableau présente les valeurs pour la plupart des périphériques et les autres rangées présentent les valeurs pour les périphériques qui font exception à la règle.

Tableau 18-10. Paramètres et gammes spécifiques au périphérique pour le VI "AI-Configurer un déclenchement" (AI Trigger Config.vi)—Partie 1

Périphérique	Type de déclenchement		Mode		Condition de déclenchement ou de pause		Niveau	
	DS*	R*	DS*	R*	DS*	R*	DS*	R*
La plupart des périphériques	2	2, 3	1	$1 \leq n \leq 3$	non supporté		non supporté	
AT-MIO-16E-10, AT-MIO-16DE-10, AT-MIO-16XE-50, PCI-MIO-16XE-50	2	$2 \leq n \leq 4$	1	$1 \leq n \leq 3$	1	1, 2, 7, 8	non supporté	

Tableau 18-10. Paramètres et gammes spécifiques au périphérique pour le VI “AI-Configurer un déclenchement” (AI Trigger Config.vi)—Partie 1 (Suite)

Périphérique	Type de déclenchement		Mode		Condition de déclenchement ou de pause		Niveau	
	DS*	R*	DS*	R*	DS*	R*	DS*	R*
AT-MIO-16E-2, AT-MIO-64E-3, NEC-MIO-16E-4	2	$1 \leq n \leq 6$	1	$1 \leq n \leq 3$	1	$1 \leq n \leq 8$	0	$-10 \leq n \leq 10$
Périphériques de Séries 1200 et Lab	2	2	1	$1 \leq n \leq 3$	non supporté		non supporté	
PC-LPM-16, DAQCard-500, DAQCard-700	non supporté		non supporté		non supporté		non supporté	
NB-A2100, NB-A2150	1	1, 2	1	$1 \leq n \leq 3$	1	1, 2	0	$-2.828 \leq n \leq 2.828$
NB-A2000	1	1, 2	1	$1 \leq n \leq 3$	1	1, 2	0	$-5.12 \leq n \leq 5.12$
Périphériques 5102	1	1, 2, 3, 6	1	$1 \leq n \leq 3$	1	1, 2, 3, 4	0	$-5 \leq n \leq 5$
5911, 5912	1	1, 2, 3, 6	1	$1 \leq n \leq 3$	1	1, 2, 3, 4	0	$-10 \leq n \leq 10$
Périphériques DSA	1	1, 2, 3	1	$1 \leq n \leq 3$	1	$1 \leq n \leq 4$	0	$-42 \leq n \leq 42$

* DS = Paramètre par défaut ; R = Gamme

Tableau 18-11. Paramètres et gammes spécifiques au périphérique pour le VI “AI-Configurer un déclenchement” (AI Trigger Config.vi)—Partie 2

Périphérique	Source de déclenchement (analogique)		Cluster de spécifications supplémentaires de déclenchement			
			Taille de la fenêtre		Couplage	
	Paramètres par défaut	Gamme	Paramètres par défaut	Gamme	Paramètres par défaut	Gamme
AT-MIO-16E-1 AT-MIO-16E-2 NEC-MIO-16E-4 PCI-MIO-16E-1 PCI-MIO-16E-1 PCI-MIO-16XE-10	0	$0 \leq n \leq 15$, PFI0	0	$0 \leq n \leq 20$	non supporté	
AT-MIO-64E-3	0	$0 \leq n \leq 63$, PFI0	0	$0 \leq n \leq 20$	non supporté	
NB-A2000	0	$0 \leq n \leq 3$	non supporté		2	1, 2
NB-A2100 NB-A2150	0	$0 \leq n \leq 3$	0	$0 \leq n \leq 5.656$	1	1, 2
Périphériques 5102	0	1, 1, TRIG	0	$0 \leq n \leq 10$	1	1, 2
PCI-6110E	0	$0 \leq n \leq 4$ PFI0	0	$0 \leq n \leq 80$	1	1, 2
PCI-6111E	0	$0 \leq n \leq 2$ PFI0	0	$0 \leq n \leq 80$	1	1, 2
4451, 4551	0	0, 1	0	$0 \leq n \leq 84$	non supporté	
4452, 4552	0	$0 \leq n \leq 4$	0	$0 \leq n \leq 84$	non supporté	
Tous les autres périphériques	non supporté		non supporté		non supporté	

Tableau 18-12. Paramètres et gammes spécifiques au périphérique pour le VI “AI-Configurer un déclenchement” (AI Trigger Config.vi)—Partie 3

Périphérique	Source de déclenchement (numérique)	
	DS	R
Déclenchement de démarrage de Série E	PFI0	PFI 0~9, RTSI 0~6, GPCTRO
Déclenchement d'arrêt de Série E	PFI1	PFI 0~9, RTSI 0~6
Fenêtrage d'horloge de balayage numérique de Série E	PFI0	PFI 0~9, RTSI 0~6
Périphériques 5102 avec RTSI et déclenchements de démarrage et d'arrêt	PFI0	PFI 1-2, RTSI 0-6
Périphériques 5102 sans RTSI et déclenchements de démarrage et d'arrêt	PFI0	PFI1-2
5911, 5912	PFI1	PFI 1-2, RTSI 0-6
Déclenchement de démarrage DSA 44xx	PFI0	PFI0, PFI1, PFI3, PFI4, PFI6, RTSI 0~6
Déclenchement d'arrêt OSA 44xx	PFI1	PFI0, PFI1, PFI3, PFI4, PFI6, RTSI 0~6
Déclenchements de démarrage et d'arrêt DSA 45xx	broche EXTTRIG* dédiée	PFI 0~33, RTSI 0~6
Tous les autres périphériques	non supporté*	

* Reportez-vous au tableau 18-9 pour la liste des périphériques pourvus de sources de déclenchement numérique fixes.

Tableau 18-13. Paramètres et gammes spécifiques au périphérique pour le VI “AI-Configurer un déclenchement” (AI Trigger Config.vi)—Partie 4

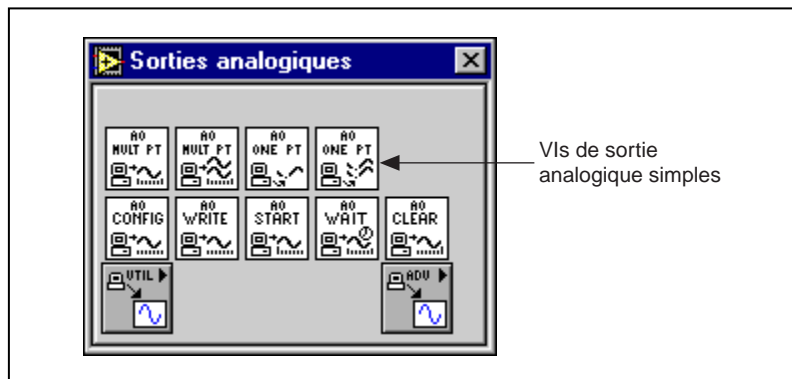
Périphérique	Cluster de spécifications de déclenchement supplémentaires					
	Délai		Sauter comptage		Limite temporelle	
	DS	R	DS	R	DS	R
NB-A2000	0	$0 \leq n \leq 655.35$	non supporté		non supporté	
NB-A2100, NB-A2150S	0	$0 \leq n \leq 32.77$	non supporté		non supporté	
NB-A2150C	0	$0 \leq n \leq 16.38$	non supporté		non supporté	
NB-A2150F	0	$0 \leq n \leq 17.05$	non supporté		non supporté	
Tous les autres périphériques	non supporté		non supporté		non supporté	

* DS = Paramètre par défaut ; R = Gamme

Vis de sorties analogiques simples

Ce chapitre décrit les VIs de sorties analogiques simples dans LabVIEW, qui exécutent des opérations simples de sorties analogiques. Vous pouvez exécuter ces VIs depuis la face-avant ou comme des sous-VIs dans des applications de base.

Vous pouvez trouver les VIs de sorties analogiques dans la palette **Fonctions»Acquisition de données»Sorties analogiques**. Les VIs de sorties analogiques simples résident dans la rangée supérieure de la palette **Sorties analogiques**, comme illustré ci-dessous.

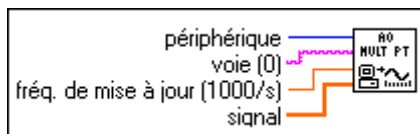


Descriptions des VIs de sorties analogiques simples

Les VIs de sorties analogiques simples suivants sont disponibles.

AO-Générer un signal (AO Generate Waveform.vi)

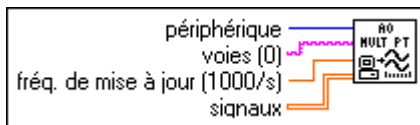
Génère un signal sur une voie de sortie analogique à la fréquence d'échantillonnage spécifiée.



Le VI “AO-Générer un signal” (AO Generate Waveform.vi) génère un signal multipoint sur la voie de sortie analogique spécifiée. Si une erreur survient, une boîte de dialogue apparaît, vous offrant le choix d’arrêter le VI ou de continuer.

AO-Générer des signaux (AO Generate Waveforms.vi)

Génère plusieurs signaux sur les voies de sorties analogiques spécifiées avec la fréquence de mise à jour spécifiée.



Si une erreur survient, une boîte de dialogue apparaît, vous offrant le choix d’interrompre l’opération ou de continuer l’exécution.

Reportez-vous à l’Annexe B, *Capacités matérielles DAQ*, pour obtenir les numéros de voies disponibles sur votre périphérique DAQ.

AO-Mettre à jour une voie (AO Update Channel.vi)

Ecrit une valeur spécifiée sur une voie de sortie analogique.



Le VI “AO-Mettre à jour une voie” (AO Update Channel.vi) écrit une valeur sur la voie de sortie analogique spécifiée. Si une erreur survient, une boîte de dialogue apparaît, vous offrant le choix d’interrompre l’opération ou de continuer l’exécution.

Reportez-vous à l’Annexe B, *Capacités matérielles DAQ*, pour obtenir les numéros de voies et les limites de sorties disponibles sur votre périphérique DAQ.

AO-Mettre à jour des voies (AO Update Channels.vi)

Ecrit des valeurs dans chacune des voies de sortie analogique spécifiées.



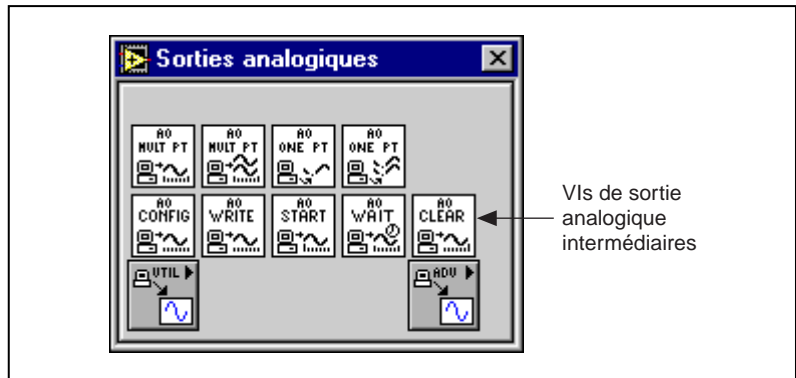
Le VI “AO-Mettre à jour des voies” (AO Update Channels.vi) effectue la mise à jour de plusieurs sorties analogiques avec des valeurs uniques. Si une erreur survient, une boîte de dialogue apparaît, vous offrant le choix d’arrêter le VI ou de continuer.

Reportez-vous à l’Annexe B, *Capacités matérielles DAQ*, pour obtenir les numéros de voies disponibles sur votre périphérique DAQ.

VIs de sorties analogiques intermédiaires

Ce chapitre décrit les VIs de sorties analogiques intermédiaires. Ces VIs sont des solutions de VIs simples à des problèmes de sorties analogiques classiques. Les VIs de niveau intermédiaire sont faciles d'emploi mais peu flexibles. Vu que tous les VIs de ce chapitre reposent sur la couche avancée, reportez-vous au chapitre 22, *VIs de sorties analogiques avancées*, pour plus d'informations sur les entrées, les sorties et leur fonctionnement.

Vous pouvez accéder aux VIs de sorties analogiques intermédiaires à partir de la palette **Fonctions»Acquisition de données»Sorties analogiques**. Les VIs de sorties analogiques intermédiaires sont dans la seconde rangée de la palette **Sorties analogiques**, comme illustré ci-dessous.



Gestion d'erreurs

Dans les VIs de sorties analogiques intermédiaires de LabVIEW, la gestion d'erreurs est simplifiée. Chaque VI de niveau intermédiaire comporte un cluster d'entrée **entrée d'erreur** et un cluster de sortie **sortie d'erreur**. Les clusters comprennent un booléen indiquant la présence ou non d'une erreur, le code d'erreur et le nom du VI qui a généré l'erreur. Si **entrée d'erreur** indique une erreur, le VI transmet les informations concernant l'erreur vers le cluster **sortie d'erreur** et arrête l'exécution.

Remarque Le VI “AO-Réinitialiser” (AO Clear.vi) est une exception à la règle. Il réinitialise toujours l’acquisition sans savoir si l’entrée d’erreur indique une erreur.

Quand vous utilisez n’importe quel VI de sorties analogiques intermédiaires dans une boucle While, arrêtez la boucle si le booléen **état** du cluster **sortie d’erreur** est Vrai (TRUE). Si vous câblez le cluster d’erreur VI “Gestionnaire général d’erreurs” (General Error Handler.vi), le VI déchiffre les informations relatives à la sortie d’erreur et vous décrit l’erreur.

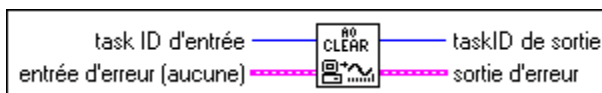
Le VI “Gestionnaire général d’erreurs” (General Error Handler.vi) se trouve dans **Fonctions»Temps et dialogue** de LabVIEW. Pour plus d’informations sur ce VI, reportez-vous à votre *Manuel de l’utilisateur LabVIEW*.

Descriptions des VIs de sorties analogiques

Les VIs de sorties analogiques suivants sont disponibles.

AO-Réinitialiser (AO Clear.vi)

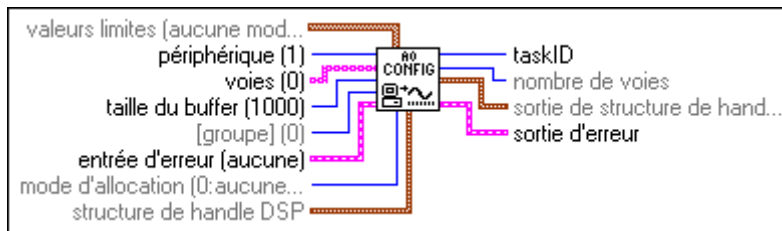
Efface la tâche de sortie analogique associée à **taskID d’entrée**.



Le VI “AO-Réinitialiser” (AO Clear.vi) réinitialise toujours la génération sans savoir si l’**entrée d’erreur** indique une erreur.

AO-Configurer (AO Config.vi)

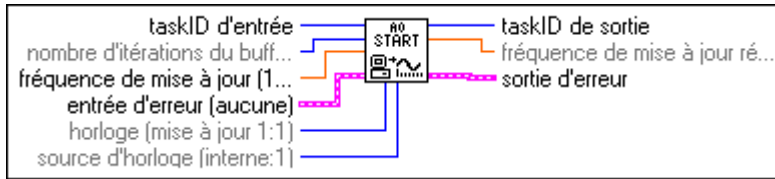
Configure la liste de voies et les limites de sorties puis alloue un buffer pour l’opération de sortie analogique.



Reportez-vous à l’Annexe B, *Capacités matérielles DAQ*, pour obtenir les gammes de voies, les limites de sorties et l’ordre de balayage disponibles sur votre périphérique DAQ.

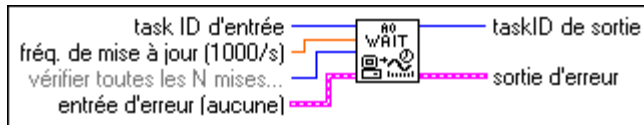
AO-Démarrer (AO Start.vi)

Démarre une opération de sortie analogique bufferisée. Ce VI définit une fréquence de mise à jour puis commence la génération.



AO-Attendre (AO Wait.vi)

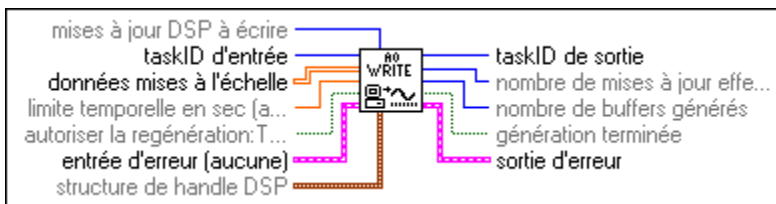
Attend que la génération de la forme d'onde s'achève.



Utilisez le VI “AO-Attendre” (AO Wait.vi) pour attendre qu’une génération finie de forme d’onde bufferisée se termine avant d’appeler le VI “AO-Réinitialiser” (AO Clear.vi). Le VI “AO-Attendre” (AO Wait.vi) vérifie l’état à intervalles réguliers en appelant le VI “AO-Ecrire” (AO Write.vi) et en vérifiant la sortie **génération terminée**. Le VI “AO-Attendre” (AO Wait.vi) attend de façon asynchrone entre intervalles et libère le processeur pour d’autres opérations. Le VI calcule l’intervalle d’attente en divisant l’entrée **vérifier toutes les N mises à jour** par la fréquence de mise à jour. N’utilisez pas le VI “AO-Attendre” (AO Wait.vi) lorsque vous générez des données en continu car la génération ne finit jamais. Le VI “AO-Réinitialiser” (AO Clear.vi) arrête une génération de forme d’onde en continu.

AO-Ecrire (AO Write.vi)

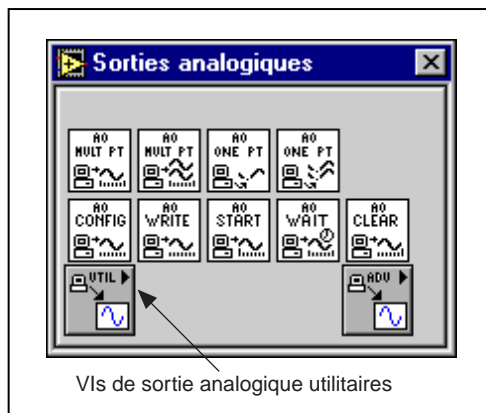
Ecrit des données dans le buffer pour une opération de sortie analogique bufferisée.



Vis Utilitaires de sorties analogiques

Ce chapitre décrit les Vis Utilitaires de sorties analogiques. Les Vis “AO-Générer en continu” (AO Continuous Gen.vi), “AO-Générer des signaux” (AO Waveform Gen.vi) et “AO-Ecrire une mise à jour” (AO Write One update.vi) sont des solutions à un seul VI apportées aux problèmes de sorties analogiques courants. Les Vis Utilitaires de sorties analogiques sont des Vis de niveau intermédiaire qui dépendent des Vis de niveau avancé. Reportez-vous au chapitre 22, *Vis de sorties analogiques avancés*, pour plus d’informations sur les entrées et les sorties et leur fonctionnement.

Vous pouvez accéder à la palette **Utilitaires de sorties analogiques** en choisissant **Fonctions»Acquisition de données»Sorties analogiques»Utilitaire de sorties analogiques**. L’icône que vous devez sélectionner pour accéder aux Vis Utilitaires de sorties analogiques se trouve sur la dernière rangée de la palette **Sorties analogiques**, comme illustré ci-dessous.



Gestion d'erreurs

Dans les VIs de sorties analogiques intermédiaires de LabVIEW, la gestion d'erreurs est simplifiée. Chaque VI de niveau intermédiaire comporte un cluster d'entrée **entrée d'erreur** et un cluster de sortie **sortie d'erreur**. Les clusters comprennent un booléen indiquant la présence ou non d'une erreur, le code d'erreur et le nom du VI qui a généré l'erreur. Si **entrée d'erreur** indique une erreur, le VI transmet les informations concernant l'erreur vers le cluster **sortie d'erreur** et arrête l'exécution.

Quand vous utilisez n'importe quel VI Utilitaire de sorties analogiques dans une boucle While, vous devez arrêter la boucle si le booléen **état** du cluster **sortie d'erreur** indique Vrai (TRUE). Si vous câblez le cluster d'erreur sur le VI "Gestionnaire général d'erreurs" (General Error Handler.vi), le VI déchiffre l'information d'erreur et vous décrit l'erreur.

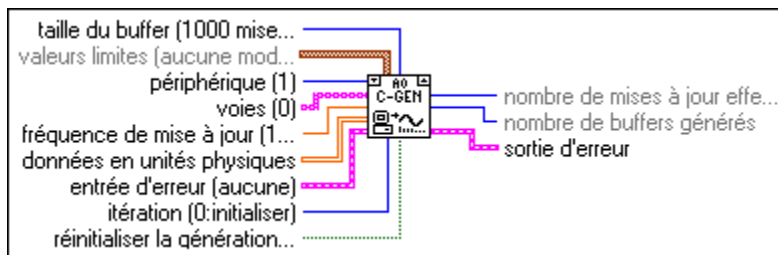
Le VI "Gestionnaire général d'erreurs" se trouve dans la palette **Fonctions»Utilitaires** de LabVIEW. Pour plus d'informations sur ce VI, reportez-vous au chapitre 10, *Fonctions de temps, de dialogue et d'erreurs*.

Descriptions des VIs Utilitaires de sorties analogiques

Les VIs Utilitaires de sorties analogiques disponibles sont les suivants.

AO-Générer en continu (AO Continuous Gen)

Génère une forme d'onde bufférisée, séquencée et en continu de façon circulaire pour les voies de sorties à la fréquence de mise à jour spécifiée. Le VI met à jour le buffer de sortie de manière continue, au fur et à mesure que les données sont générées. Si vous voulez simplement générer les mêmes données sans discontinuer, utilisez plutôt le VI "AO-Générer des signaux" (AO Waveform Gen.vi).





Utilisez le VI “AO-Générer en continu” (AO Continuous Gen.vi) quand les données de votre signal résident sur disque et sont trop volumineuses pour la mémoire ou quand vous devez créer votre forme d’onde en temps réel. Placez le VI dans une boucle While et câblez le terminal d’itération à l’entrée **itération** du VI.



Remarque *Si votre boucle s’effectue plus de $2^{31}-1$ fois, ne câblez pas ce terminal d’itération du VI au terminal d’itération de boucle. Positionnez plutôt l’entrée itération sur 0 pour la première itération de la boucle, puis sur n’importe quelle valeur positive pour toutes les autres itérations. Le VI reconfigure et redémarre si itération ≤ 0 .*

Câblez également la condition qui arrête la boucle à l’entrée **réinitialiser l’acquisition** du VI, en inversant le signal, si nécessaire, pour obtenir Vrai (TRUE) à la dernière itération. Lors de l’itération 0, le VI appelle le VI “AO-Configurer” (AO Config.vi) pour configurer le groupe de voies ainsi que le matériel et pour allouer un buffer aux données. Il appelle également le VI “AO-Ecrire” (AO Write.vi) pour écrire les données en question dans le buffer puis le VI “AO-Démarrer” (AO Start.vi) pour régler la fréquence de mise à jour et démarrer la génération de signal. Ensuite, à chaque itération, le VI appelle le VI “AO-Ecrire” (AO Write.vi) pour écrire la partie de données suivante dans le buffer à la position d’écriture courante. Lors de la dernière itération [(quand **réinitialiser la génération** est Vrai (TRUE)] ou si une erreur survient, le VI appelle également le VI “AO-Réinitialiser” (AO Clear.vi) pour réinitialiser toute génération en cours. Même si normalement cela n’est pas nécessaire, vous pouvez appeler le VI “AO-Générer en continu” (AO Continuous Gen.vi) à l’extérieur d’une boucle (c’est-à-dire, pour ne l’appeler qu’une seule fois). Mais ce faisant, ne câblez pas les entrées **itération** et **réinitialiser la génération**.

Le premier appel au VI “AO-Ecrire” (Write) positionne le paramètre **autoriser la régénération** sur Vrai (TRUE), pour que les mêmes données soient générées plusieurs fois. Si vous définissez **autoriser la régénération** sur Faux (FALSE), vous devez écrire de nouvelles données suffisamment rapidement pour qu’elles soient toujours prêtes à être générées. Si vous ne remplissez pas le buffer suffisamment rapidement, une erreur de génération peut survenir. Pour pallier à ce problème, diminuez la **fréquence de mise à jour**, augmentez la **taille du buffer**, augmentez le volume de données écrites chaque fois ou bien écrivez des données plus souvent.

Si vous définissez **autoriser la régénération** sur Faux (FALSE), et que votre périphérique dispose d’une FIFO de sorties analogiques, la **taille du buffer** doit être deux fois plus grande que votre FIFO.

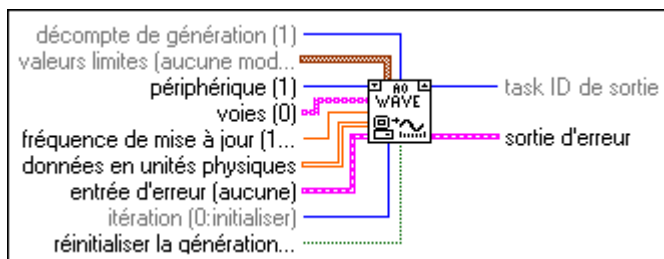
Si une erreur survient, le VI appelle le VI “AO-Réinitialiser” (AO Clear.vi) pour effacer toute génération en cours, puis passe l’information d’erreur non modifiée au cluster **sortie d’erreur**. Si une erreur survient à l’intérieur du VI “AO-Générer en continu” (Continuous Gen.vi), le VI “AO-Réinitialiser” (AO Clear.vi) efface toute génération en cours et transmet les informations sur cette erreur.

Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour connaître les gammes de voies et les limites de sortie disponibles sur votre périphérique DAQ.

Remarque *Le VI "AO-Générer en continu" (AO Continuous Gen.vi) utilise un registre à décalage non initialisé comme mémoire locale pour ne pas oublier le taskID de l'opération de sortie entre les appels. Vous utilisez normalement ce VI à un seul endroit d'un diagramme, mais si vous l'utilisez à plusieurs endroits, les instances multiples du VI partagent le même taskID. Tous les appels à ce VI configurent, écrivent des données ou réinitialisent la même génération. Si vous le souhaitez, vous pouvez utiliser ce VI à plusieurs endroits du diagramme mais chaque instance doit se référer à un taskID différent (par exemple, quand vous voulez générer des formes d'onde avec deux périphériques simultanément). Sauvegardez une copie de ce VI sous un nouveau nom (par exemple, "AO-Générer en continu R" [AO Continuous Gen R.vi]) et rendez votre VI ré-entrant.*

AO-Générer des signaux (AO Waveform Gen)

Génère une forme d'onde séquencée, bufférisée simple ou de façon circulaire pour les voies de sortie à la fréquence de mise à jour spécifiée. Sauf si vous exécutez une génération indéfinie, le VI ne redonne la main au diagramme de LabVIEW qu'une fois la génération terminée.



terminal
d'itération

Si vous placez ce VI dans une boucle pour générer des signaux multiples avec le même groupe de voies, câblez le terminal d'itération à l'entrée **itération** du VI.

Remarque *Si votre boucle s'effectue plus de $2^{31} - 1$ fois, ne câblez pas ce terminal d'itération du VI au terminal d'itération de boucle. Au lieu de cela, positionnez l'entrée itération sur 0 pour la première itération de la boucle, puis sur n'importe quelle valeur positive pour toutes les autres itérations. Le VI reconfigure et redémarre si itération ≤ 0 .*

Lors de l'itération 0, le VI appelle le VI "AO-Configurer" (AO Config.vi) pour configurer le groupe de voies ainsi que le matériel et pour allouer un buffer aux données. A chaque itération, le VI appelle le VI "AO-Ecrire" (AO Write.vi) pour écrire les données dans le buffer

puis le VI “AO-Démarrer” (AO Start.vi) pour régler la fréquence de mise à jour et démarrer la génération. Si vous n’appellez le VI “AO-Générer des signaux” (AO Waveform Gen.vi) qu’une seule fois, vous n’avez pas besoin de câbler l’entrée **itération**. Le paramètre par défaut de **itération** est 0, indiquant ainsi au VI qu’il faut configurer le périphérique avant de démarrer la génération de signaux.

Si une erreur survient, le VI appelle le VI “AO-Réinitialiser” (AO Clear.vi) pour réinitialiser toute génération en cours, puis passe l’information d’erreur non modifiée par **sortie d’erreur**. Si une erreur survient à l’intérieur du VI “AO-Générer des signaux (AO Waveform Gen.vi), il réinitialise toute génération en cours et transmet les informations sur cette erreur.

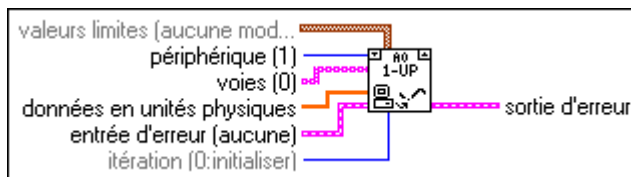
Reportez-vous à l’Annexe B, *Capacités matérielles DAQ*, pour les gammes de données, les limites de sortie et l’ordre de balayage disponibles sur votre périphérique DAQ.



Remarque *Le VI “AO- Générer des signaux” (AO Waveform Gen.vi) utilise un registre à décalage non initialisé comme mémoire locale pour ne pas oublier le taskID de l’opération de sortie entre les appels. Vous utilisez normalement ce VI à un seul endroit d’un diagramme, mais si vous l’utilisez à plusieurs endroits, les instances multiples du VI partagent le même taskID. Tous les appels à ce VI configurent, écrivent des données ou réinitialisent la même génération. Si vous le souhaitez, vous pouvez utiliser ce VI à plusieurs endroits du diagramme mais chaque instance doit se référer à un taskID différent (par exemple, quand vous voulez générer des signaux avec deux périphériques simultanément). Sauvegardez une copie de ce VI sous un nouveau nom (par exemple, “AO-Générer en continu R”) et rendez votre VI ré-entrant.*


AO-Ecrire une mise à jour (AO Write One Update.vi)

Ecrit une valeur unique pour chacune des voies de sorties analogiques spécifiée.




terminal
d’itération

Le VI “AO-Ecrire une mise à jour” (AO Write One Update.vi) effectue une mise à jour non séquentielle et immédiate d’un groupe d’une ou de plusieurs voies. Si vous placez le VI dans une boucle pour écrire plusieurs valeurs pour le même groupe de voies, câblez le terminal d’itération à l’entrée **itération** du VI.

 **Remarque** *Si votre boucle s’effectue plus de $2^{31}-1$ fois, ne câblez pas ce terminal d’itération du VI au terminal d’itération de boucle. Au lieu de cela, positionnez l’entrée itération sur 0 sur la première itération de la boucle, puis sur n’importe quelle valeur positive pour toutes les autres itérations. Le VI reconfigure et redémarre si itération ≤ 0 .*

Lors de l’itération 0, le VI appelle le VI “AO-Configurer” (AO Config.vi) pour configurer le groupe de voies ainsi que le matériel, puis appelle le VI “AO-Mettre à jour une seule fois” (AO Single Update.vi), pour écrire la tension sur les sorties. Lors des itérations suivantes, le VI n’appelle que le VI “AO-Mettre à jour une seule fois” (AO Single Update.vi), évitant ainsi une configuration inutile. Si vous n’appelez le VI “AO-Ecrire une mise à jour” (AO Write One Update.vi) qu’une seule fois pour écrire une valeur unique pour chaque voie, ne câblez pas l’entrée **itération**. Sa valeur par défaut égale à 0 indique au VI d’effectuer une configuration avant d’écrire les données.

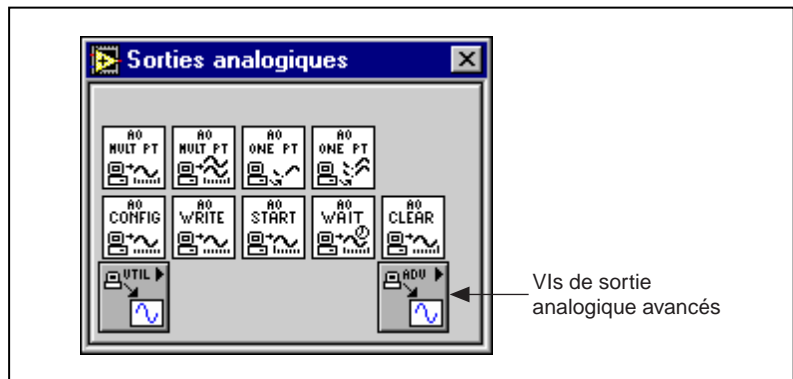
Reportez-vous à l’Annexe B, *Capacités matérielles DAQ*, pour les gammes de données, les limites de sortie et l’ordre de balayage disponibles sur votre périphérique DAQ.

 **Remarque** *Le VI “AO-Ecrire une mise à jour” (AO Write One Update.vi) utilise un registre à décalage non initialisé comme mémoire locale pour ne pas oublier le taskID du groupe de voies lors des appels entre VIs. Habituellement, ce VI apparaît à un seul endroit de votre diagramme. Toutefois, si vous l’utilisez à plusieurs endroits, les instances multiples du VI partagent le même taskID. Tous les appels à ce VI configurent ou écrivent des données pour ce même groupe. Si vous voulez utiliser ce VI à plusieurs endroits de votre diagramme et que chaque instance a un taskID différent (par exemple, pour écrire des données avec deux périphériques en même temps), vous devez sauvegarder une copie de ce VI sous un nouveau nom (par exemple, “AO-Ecrire une mise à jour R” [AO Write One Update R.vi]) et rendez votre VI ré-entrant.*

Vis de sorties analogiques avancés

Ce chapitre contient des descriptions de références des VIs de sorties analogiques avancés. Ces VIs constituent une interface avec le logiciel NI-DAQ et la base des VIs de sorties analogiques intermédiaires, utilitaires et simples.

Vous pouvez accéder à la palette **Sorties analogiques avancées** en choisissant **Fonctions»Acquisition de données»Sorties analogiques»Sorties analogiques avancées**. L'icône que vous devez sélectionner pour accéder aux VIs de sorties analogiques avancés se trouve sur la rangée du bas de la palette **Sorties analogiques**, comme illustré ci-dessous.



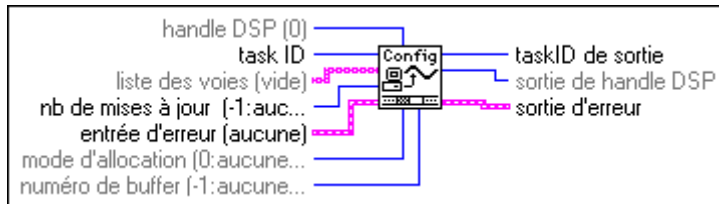
Descriptions des VIs de sorties analogiques avancés

Les VIs de sorties analogiques avancés suivants sont disponibles.

AO-Configurer un buffer (AO Buffer Config.vi)

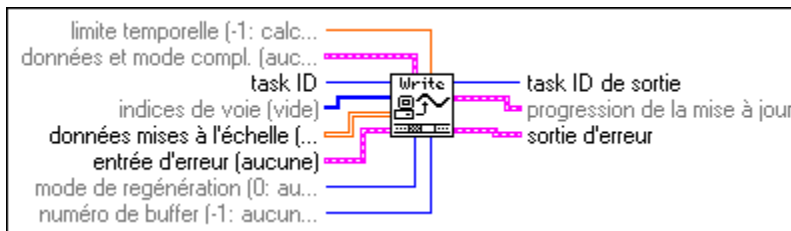
Alloue de la mémoire pour un buffer de sorties analogiques. Si vous utilisez des interruptions, vous pouvez allouer une série de buffers de sorties analogiques et les affecter à un groupe en appelant le VI “AO-Configurer un buffer” (AO Buffer Config.vi) plusieurs fois. Chaque buffer peut avoir sa propre taille. Si vous utilisez le mode DMA, vous ne pouvez allouer qu’un seul buffer.

Utilisez le numéro que vous affectez au buffer avec ce VI lorsque vous avez besoin de vous référer à ce buffer avec d’autres VIs.



AO-Ecrire dans un buffer (AO Buffer Write.vi)

Ecrit des données de sorties analogiques dans des buffers créés par le VI “AO-Configurer un buffer” (AO Buffer Config.vi).

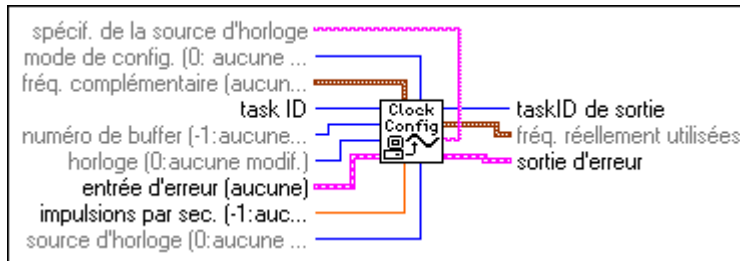


Vous câblez les nouvelles données à générer sur l’une des trois entrées : **données en unités physiques**, **données binaires**, ou **handle de mémoire DSP**. Le VI recherche ces entrées dans cet ordre pour le premier tableau d’une longueur supérieure à zéro. Puis le VI écrit les données provenant de ce tableau dans le buffer de sortie. La longueur des tableaux de **données en unités physiques** ou de **données binaires** détermine le nombre de mises à jour que le VI effectue. Si **handle de mémoire DSP** pointe vers la source des données, **mises à jour à écrire** doit indiquer combien de mises à jour le VI doit écrire. Lorsqu’aucune des données n’est câblée, ce VI peut être utile pour reporter des informations sur la progression de la mise à jour.

Le nombre total de mises à jour écrites dans un buffer avant le démarrage, peut être inférieur au nombre de mises à jour attribuées au buffer quand vous avez appelé le VI “AO-Configurer un buffer” (AO Buffer Config.vi). LabVIEW ne génère alors que les mises à jour écrites dans le buffer.

AO-Configurer une horloge (AO Clock Config.vi)

Configure une horloge d'intervalle ou de mise à jour pour sorties analogiques.



Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour connaître les horloges disponibles sur votre périphérique DAQ.

Vous pouvez exprimer des fréquences d'horloge de trois façons : en **impulsions par seconde**, **secondes par impulsion** ou par les trois paramètres de temps réel. Le VI recherche ces paramètres dans cet ordre et exprime les fréquences d'horloge sur le premier paramètre avec une entrée valide câblée. Lorsque vous configurez une horloge de mise à jour, une impulsion correspond à une mise à jour. Lorsque vous configurez l'horloge d'intervalle, une impulsion correspond à un intervalle.

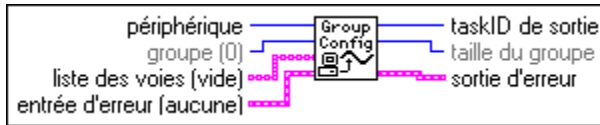
AO-Contrôler (AO Control.vi)

Démarre, interrompt, reprend et réinitialise les tâches de sorties analogiques.



AO-Configurer un groupe (AO Group Config.vi)

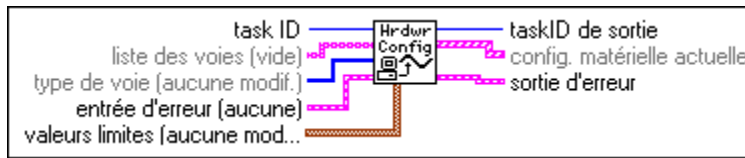
Affecte une liste de voies de sorties analogiques à un numéro de groupe et fournit le taskID que tous les autres VIs de sorties analogiques utilisent.



Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour connaître les voies de votre périphérique DAQ disponibles.

AO-Configurer un matériel (AO Hardware Config.vi)

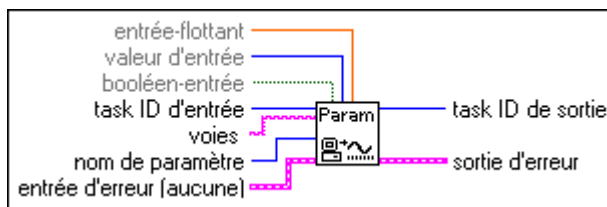
Configure les limites (polarité et référence) et détermine si les données sur une voie sont exprimées en volts ou milliampères, si vous utilisez des numéros de voies. Ce VI retourne toujours les paramètres réels pour toutes les voies du groupe.



Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour connaître les gammes des voies et les limites de sortie de votre périphérique DAQ disponibles.

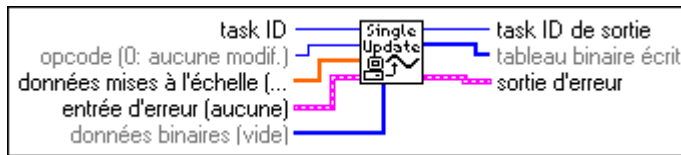
AO-Paramètre (AO Parameter.vi)

Règle les divers paramètres associés à l'opération de sorties analogiques des périphériques qui ne sont pas couverts par d'autres VIs de sorties analogiques.



AO-Mettre à jour une seule fois (AO Single Update.vi)

Exécute une mise à jour immédiate des voies du groupe.



AO-Configurer un déclenchement et un fenêtrage (AO Trigger and Gate Config.vi) (Windows)

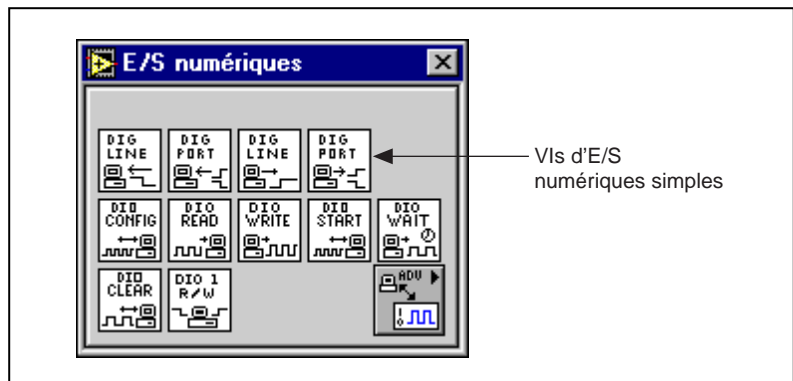
Configure les conditions de déclenchement et de fenêtrage pour les opérations de sorties analogiques sur des périphériques de Série E et 5411.



Vis d'E/S numériques simples

Ce chapitre décrit les VIs qui exécutent des opérations d'E/S numériques simples. Vous pouvez exécuter ces VIs à partir de la face-avant ou bien les utiliser comme sous-VIs dans des applications de base.

Vous pouvez accéder aux VIs d'E/S numériques simples en choisissant la palette **Fonctions**»**Acquisition de données**»**E/S numériques**.



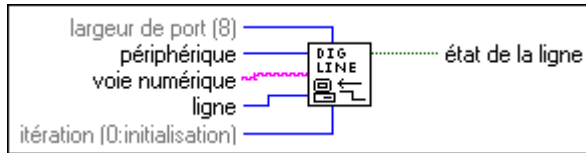
Les VIs d'E/S numériques simples se trouvent sur la rangée du haut de la palette **E/S numériques**. Pour consulter des exemples sur la façon d'utiliser ces VIs, ouvrez la bibliothèque `examples\daq\digital\digital.llb`.

Description des VIs d'E/S numériques simples

Les VIs d'E/S numériques simples suivants sont disponibles.

Lire une ligne numérique (Read from Digital Line.vi)

Lit l'état logique d'une ligne numérique sur une voie numérique que vous configurez.



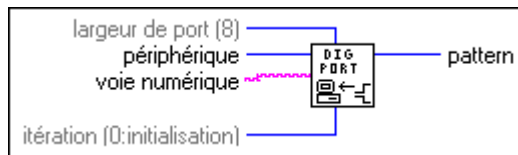
Si une erreur survient, une boîte de dialogue apparaît vous donnant le choix entre arrêter le VI ou continuer l'exécution.



Remarque *Lorsque vous appelez ce VI sur un port d'E/S numériques qui fait partie d'un 8255 PPI et que votre terminal d'itération est laissé à 0, le 8255 PPI passe par une phase de configuration, où tous les ports d'un même circuit PPI sont mis à l'état logique bas indépendamment de la direction. Toutefois, la direction des données sur d'autres ports est maintenue. Pour éviter cet effet, connectez une valeur différente de 0 au terminal d'itération une fois que vous avez configuré les ports désirés.*

Lire un port numérique (Read from Digital Port.vi)

Lit une voie numérique que vous configurez.



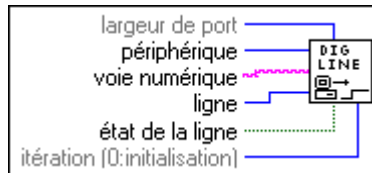
Si une erreur survient, une boîte de dialogue apparaît vous donnant le choix entre arrêter le VI ou continuer l'exécution.



Remarque *Lorsque vous appelez ce VI sur un port d'E/S numériques qui fait partie d'un 8255 PPI et que votre terminal d'itération est laissé à 0, le 8255 PPI passe par une phase de configuration, où tous les ports d'un même circuit PPI sont mis à l'état logique bas indépendamment de la direction. Pour éviter cet effet, connectez une valeur différente de 0 au terminal d'itération une fois que vous avez configuré les ports désirés.*

Ecrire sur une ligne numérique (Write to Digital Line.vi)

Règle l'état logique d'une ligne numérique de sortie sur un état haut ou bas que vous spécifiez.

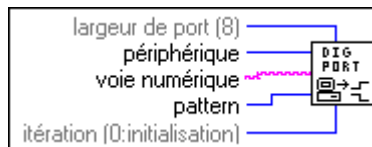


Si une erreur survient, une boîte de dialogue apparaît vous donnant le choix entre arrêter le VI ou continuer l'exécution.

Remarque *Lorsque vous appelez ce VI sur un port d'E/S numériques qui fait partie d'un 8255 PPI et que votre terminal d'itération est laissé à 0, le 8255 PPI passe par une phase de configuration, où tous les ports d'un même circuit PPI sont mis à l'état logique bas indépendamment de la direction. Toutefois, la direction des données sur d'autres ports est maintenue. Pour éviter cet effet, connectez une valeur différente de 0 au terminal d'itération une fois que vous avez configuré les ports désirés.*

Ecrire sur un port numérique (Write to Digital Port.vi)

Sort un modèle numérique pour une voie numérique que vous spécifiez.



Si une erreur survient, une boîte de dialogue apparaît vous donnant le choix entre arrêter le VI ou continuer l'exécution.



Remarque *Lorsque vous appelez ce VI sur un port d'E/S numériques qui fait partie d'un 8255 PPI et que votre terminal d'itération est laissé à 0, le 8255 PPI passe par une phase de configuration, où tous les ports d'un même circuit PPI sont mis à l'état logique bas indépendamment de la direction. Toutefois, la direction des données sur d'autres ports est maintenue. Pour éviter cet effet, connectez une valeur différente de 0 au terminal d'itération une fois que vous avez configuré les ports désirés.*

VIs d'E/S numériques intermédiaires

Ce chapitre décrit les VIs d'E/S numériques intermédiaires. Ces VIs sont des solutions simples à des problèmes numériques classiques.

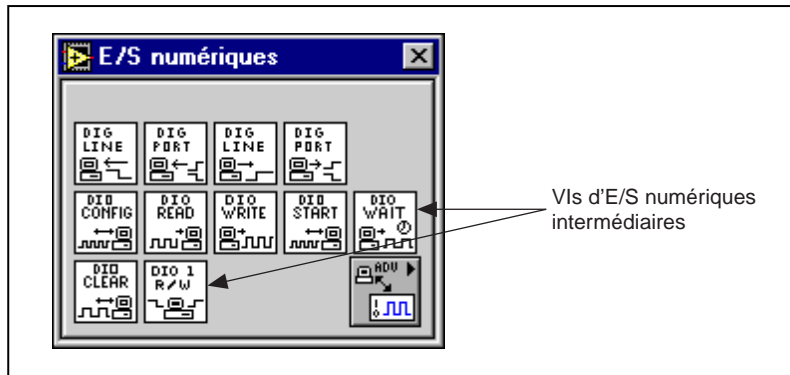
Par exemple, le VI “DIO-Lire et écrire une fois” (DIO Single Read/Write.vi) est une solution simple pour des lectures et des écritures numériques non bufférisées. Le VI “DIO-Lire/écrire une fois” (DIO Single Read/Write.vi) fonctionne avec n'importe quel périphérique doté de ports numériques.

Vous combinez les autres VIs “DIO-Configurer” (DIO Config.vi), “DIO-Démarrer” (DIO Start.vi), “DIO-Lire” (DIO Read.vi), “DIO-Ecrire” (DIO Write.vi), “DIO-Attendre” (DIO Wait.vi) et “DIO-Réinitialiser” (DIO Clear.vi) pour construire des applications plus exigeantes utilisant des lectures et des écritures bufférisées. Votre périphérique doit supporter le handshaking pour utiliser ces VIs.

Tous les VIs décrits dans ce chapitre sont construits à partir de la couche de blocs de construction fondamentale, les VIs de niveau avancé.

Vous pouvez accéder aux VIs d'E/S numériques intermédiaires en choisissant la palette **Fonctions»Acquisition de données»E/S numériques**. Les VIs d'E/S numériques intermédiaires sont les VIs des

deuxième et troisième rangées de la palette **E/S Numériques**, comme illustré ci-dessous.



Gestion d'erreurs

Dans les VIs d'entrées/sorties numériques intermédiaires de LabVIEW, la gestion d'erreurs est simplifiée. Chaque VI de niveau intermédiaire comporte un cluster d'entrée **entrée d'erreur** et un cluster de sortie **sortie d'erreur**. Les clusters comprennent un booléen qui indique la présence d'une erreur, le code d'erreur et le nom du VI qui a retourné l'erreur. Si **entrée d'erreur** indique une erreur, le VI transmet les informations vers le cluster **sortie d'erreur** et arrête l'exécution.



Remarque *Les VIs DIO-Réinitialiser sont une exception à cette règle ; en effet, ces VIs réinitialisent toujours l'acquisition même si le cluster entrée d'erreur indique une erreur.*

Quand vous utilisez n'importe quel VI d'E/S numériques intermédiaires dans une boucle While, vous devez arrêter la boucle si le booléen **état** du cluster **sortie d'erreur** indique Vrai (TRUE). Si vous câblez le cluster d'erreur sur le VI "Gestionnaire général d'erreurs" (General Error Handler.vi), le VI déchiffre l'information d'erreur et vous décrit l'erreur.

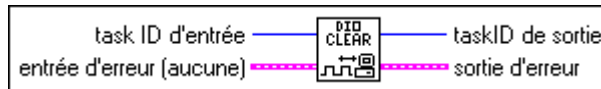
Le VI Gestionnaire général d'erreurs se trouve dans **Fonctions»Utilitaires** dans LabVIEW. Pour plus d'informations sur ce VI, reportez-vous au chapitre 10, *Fonctions de temps, de dialogue et d'erreurs*.

Descriptions des VIs d'E/S numériques intermédiaires

Les VIs d'E/S numériques intermédiaires suivants sont disponibles.

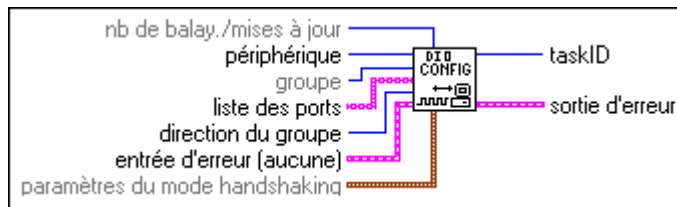
DIO-Réinitialiser (DIO Clear.vi)

Appelle le VI “Contrôler un buffer de groupe numérique” (Digital Group Buffer Control.vi) pour arrêter un transfert et réinitialiser le groupe.



DIO-Configurer (DIO Config.vi)

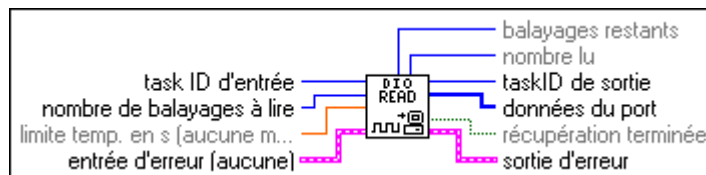
Le VI “DIO-Configurer” (DIO Config.vi) appelle le VI “Configurer un groupe numérique” (Digital Group Config.vi), pour assigner une liste de ports au groupe, établir la direction du groupe et créer un **taskID**. Le VI appelle ensuite le VI “Configurer un mode numérique” (Digital Mode Config.vi) pour établir les paramètres du handshaking, qui n’affectent que l’opération des périphériques DIO-32. Finalement, le VI appelle le VI “Configurer un buffer numérique” (Digital Buffer Config.vi) pour allouer un buffer afin de retenir les balayages pendant leur lecture ou pour permettre l’écriture de mises à jour.



Reportez-vous à l’Annexe B, *Capacités matérielles DAQ*, pour connaître les ports et les directions disponibles sur votre périphérique DAQ.

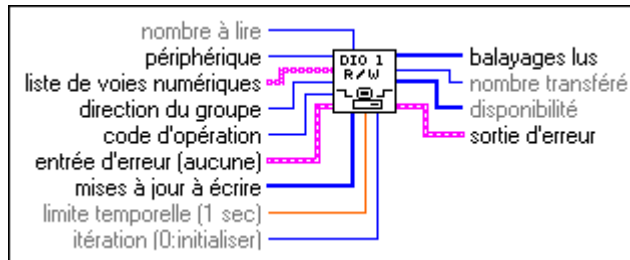
DIO-Lire (DIO Read.vi)

Appelle le VI “Lire un buffer numérique” (Digital Buffer Read.vi) pour lire les données provenant du buffer de transfert interne et retourne les données lues dans **pattern**.



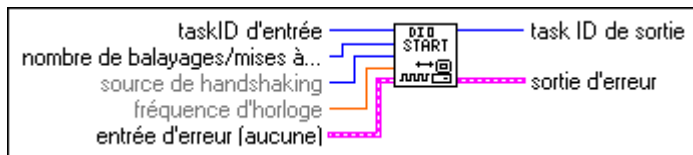
DIO-Lire/écrire une fois (DIO Single Read/Write.vi)

Lit ou écrit des données numériques sur les voies numériques spécifiées dans la liste des voies numériques. Ce VI configure et transfère des données une fois. Quand vous utilisez ce VI dans une boucle, câblez le terminal d'**itération** de la boucle à l'entrée **itération** pour que la configuration n'ait lieu qu'une fois.



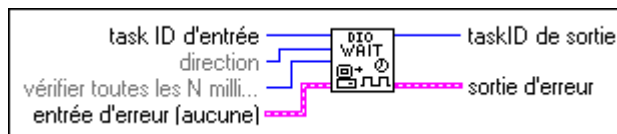
DIO-Démarrer (DIO Start.vi)

Démarre une opération d'E/S numériques bufférisée. Ce VI appelle le VI "Configurer une horloge numérique" (Digital Clock Config.vi) pour régler la fréquence d'horloge si l'horloge interne émet les signaux de handshaking, puis démarre le transfert des données en appelant le VI "Contrôler un buffer numérique" (Digital Buffer Control.vi).



DIO-Attendre (DIO Wait.vi)

Attend l'achèvement des opérations de sorties ou d'entrées numériques bufférisées. Pour l'entrée, le VI détecte l'achèvement quand l'état d'acquisition retourné par le VI "Lire un buffer numérique" (Digital Buffer Read.vi) finit avec ou sans données non traitées. Pour la sortie, le VI détecte l'achèvement lorsque l'indicateur **génération terminée** du VI "DIO-Ecrire" (DIO Write.vi) indique Vrai (TRUE).

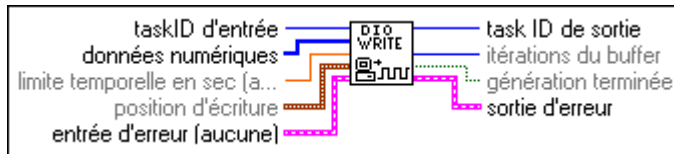


Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour connaître les modes de handshaking disponibles sur votre périphérique DAQ.

DIO-Ecrire (DIO Write.vi)

Appelle le VI "Ecrire dans un buffer numérique" (Digital Buffer Write.vi) pour écrire dans le buffer de transfert interne.

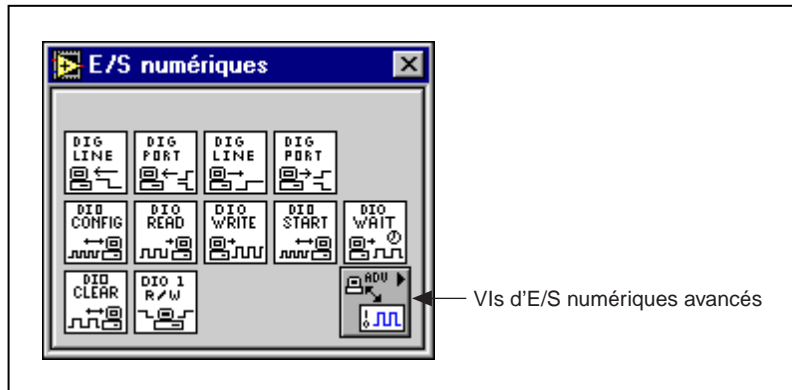
(Macintosh) Remplissez le buffer de données avant d'utiliser le VI "DIO-Démarrer" (DIO Start.vi) pour commencer l'opération de sortie numérique. Appelez le VI "DIO-Ecrire" (DIO Write.vi) après le début du transfert pour récupérer les informations d'état.



Vis d'E/S numériques avancés

Ce chapitre décrit les VIs d'E/S numériques avancés, qui incluent les VIs de groupes et de ports numériques. Vous pouvez utiliser les VIs de ports numériques pour des lectures et des écritures immédiates sur des ports et des lignes numériques. Les VIs de groupe numérique sont en général utilisés pour des E/S séquencées, immédiates ou utilisant le mode handshaking pour des ports multiples. Ces VIs constituent une interface avec le logiciel NI-DAQ et le fondement des VIs d'E/S numériques simples et intermédiaires.

Pour accéder à la palette **E/S numériques avancées**, sélectionnez **Fonctions»Acquisition de données»E/S numériques»E/S numériques avancées**. L'icône que vous devez sélectionner pour accéder aux VIs d'E/S numériques avancées se trouve sur le rang du bas de la palette **E/S numériques**, comme illustré ci-dessous.

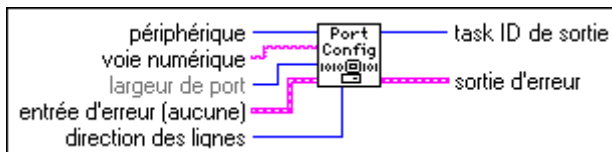


Descriptions des VIs de port numérique

Les VIs de port numérique n'effectuent que des lectures et des écritures numériques immédiates.

DIO-Configurer un port (DIO Port Config.vi)

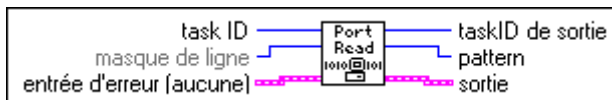
Etablit une configuration de voie numérique. Vous pouvez utiliser le **taskID** que ce VI retourne uniquement dans les VIs de port numérique.



Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour connaître les ports et les directions disponibles sur votre périphérique DAQ.

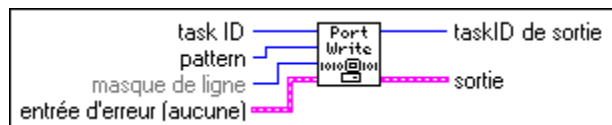
DIO-Lire un port (DIO Port Read.vi)

Lit la voie numérique identifiée par le **taskID** et retourne le modèle lu dans **pattern**.



DIO-Ecrire sur un port (DIO Port Write.vi)

Ecrit la valeur du **pattern** dans le port identifié par le **taskID**.

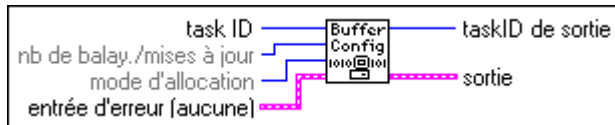


Descriptions des VIs de groupes numériques

Les VIs de groupes numériques effectuent des E/S numériques séquencées, immédiates ou utilisant le mode handshaking.

Configurer un buffer numérique (Digital Buffer Config.vi)

Alloue de la mémoire pour un buffer de sortie ou d'entrée numérique.



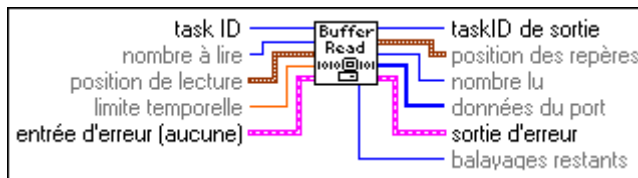
Contrôler un buffer numérique (Digital Buffer Control.vi)

Démarré une opération de sortie ou d'entrée.



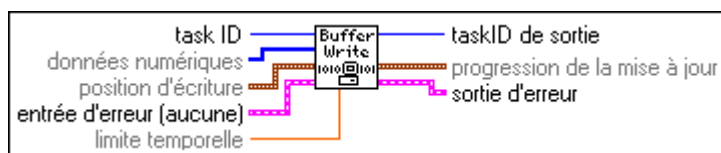
Lire un buffer numérique (Digital Buffer Read.vi)

Retourne les données d'entrées numériques situées dans le buffer de données interne.



Ecrire un buffer numérique (Digital Buffer Write.vi)

Ecrit les données de sortie numérique dans le buffer créé par le VI "Configurer un buffer numérique" (Digital Buffer Config.vi). L'écriture commence toujours à partir d'un repère d'écriture. Après une écriture, le repère d'écriture pointe vers la donnée (ou mise à jour) qui suit la dernière mise à jour écrite.

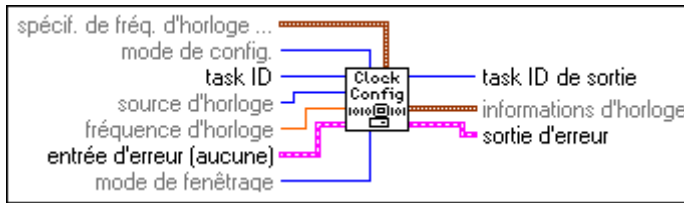


(Macintosh) Remplissez le buffer de données avant d'utiliser le VI "Contrôler un buffer numérique" (Digital Buffer Control.vi) pour commencer l'opération de sortie numérique. Vous pouvez appeler le VI "Ecrire un buffer numérique" (Digital Buffer Write.vi) après le début du transfert pour récupérer des informations d'état.

Le nombre total de mises à jour écrites dans un buffer avant de commencer l'opération, peut être inférieur au nombre de mises à jour que vous avez allouées pour le buffer quand vous avez appelé le VI "Configurer un buffer numérique" (Digital Buffer Config.vi). Le VI ne génère alors que les mises à jour écrites dans le buffer.

Configurer une horloge numérique (Digital Clock Config.vi)

Configure un périphérique DIO-32 pour émettre des signaux de handshaking basés sur la sortie d'une horloge pour E/S numériques séquencées.



Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour connaître les horloges disponibles sur votre périphérique DAQ.

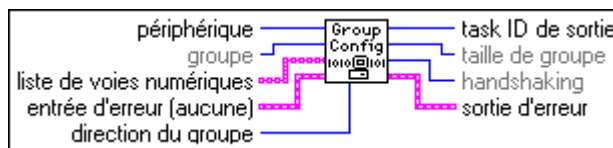
L'exemple suivant illustre la façon d'utiliser les trois paramètres de base de temps pour spécifier une fréquence d'horloge. Supposons que ces paramètres ont les réglages suivants :

- source de base de temps : 1
- signal de base de temps : 1 000 000,0 Hz
- séparateur de base de temps : 25

Dans ce cas, la fréquence d'impulsions par seconde est 1 000 000,0 divisé par 25, donc LabVIEW met à jour le groupe numérique 40 000 fois par seconde.

Configurer un groupe numérique (Digital Group Config.vi)

Définit un groupe de sortie ou d'entrée numérique. Vous ne pouvez utiliser le **taskID** que ce VI retourne que dans les VIs de groupes numériques.



Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour connaître les ports et les directions disponibles sur votre périphérique DAQ.



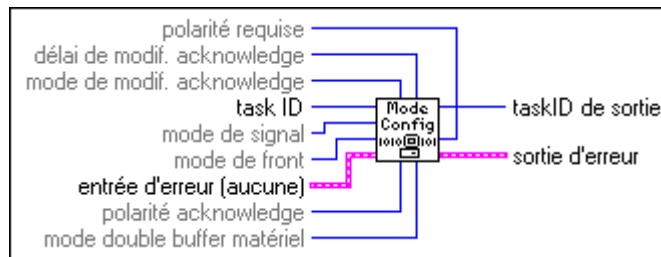
Remarque *La même voie numérique ne peut pas appartenir à deux groupes différents. Si vous configurez un groupe pour utiliser une voie numérique spécifiée, cette voie numérique ne doit pas être déjà définie dans un autre groupe ; sinon, une erreur est générée.*

Les cartes MIO (sauf l'AT-MIO-16D et l'AT-MIO-16DE-10), ainsi que les périphériques NB-TIO-10, LPM, DAQCard-500, 516, DAQCard-700, PC-TIO-10, AO-2DC, PC-OPDIO-16 et AT-AO-6/10, ne permettent pas le handshaking. Les VIs de ports numériques sont mieux appropriés pour ces périphériques. Le handshaking n'est pas permis si la **liste de voies numériques** est composée de noms de voies. L'AT-MIO-16D et le AT-MIO-16DE-10 ne permettent pas le handshaking si la **liste de voies numériques** inclut les ports 0, 1 et/ou 4. Les périphériques DIO-96 ne permettent pas le handshaking si la **liste de voies numériques** inclut les ports 2, 5, 8, et/ou 11. Les périphériques DIO-24, Lab et Série 1200 ne permettent pas le handshaking si la **liste de voies numériques** inclut le port 2. Le DIO-32F permet le handshaking pour les configurations suivantes uniquement :

- Un groupe contenant n'importe quel port
- Un groupe contenant les ports 0 et 1, ou les ports 2 et 3, dans cet ordre
- Un groupe contenant les ports 0, 1, 2 et 3, dans cet ordre

Configurer un mode numérique (Digital Mode Config.vi)

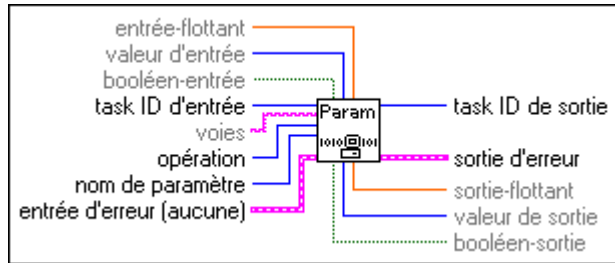
Configure les caractéristiques du handshaking pour les périphériques DIO-32.



Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour connaître les modes de handshaking disponibles sur votre périphérique DAQ.

DIO-Paramètre (DIO Parameter.vi)

Configure et récupère les paramètres divers associés à des sorties et entrées numériques qui ne sont pas configurées par d'autres VIs DIO.



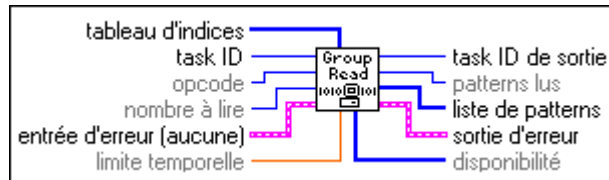
Le tableau 25-1 liste les paramètres spécifiques de périphérique et les gammes autorisées pour périphériques.

Tableau 25-1. Paramètres et valeurs autorisées spécifiques aux périphériques

Périphérique	Nom du paramètre	Support	Réglage possible	Entrée/sortie à utiliser	Valeurs autorisées	Valeurs par défaut
VXI-DI O-128	0: Entrée Seuil logique de port	port par entrée	oui	voies, flottant interne, flottant externe	Numérique/Analogique	Numérique/Analogique
DAQ-DI O-653 (DIO-32HS)	1: Echange ACK/Req	par groupe	oui	taskID interne, valeur interne, valeur externe	Arrêt, Marche	Numérique/Analogique
	2: Inverser horloge	par groupe	oui	taskID interne, valeur interne, valeur externe	Arrêt, Marche	Numérique/Analogique

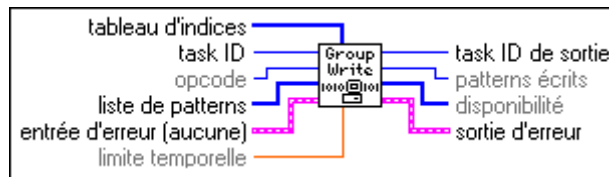
Lire une seule valeur numérique (Digital Single Read.vi)

Lit les voies numériques appartenant au groupe identifié par **taskID** et retourne les patterns lus.



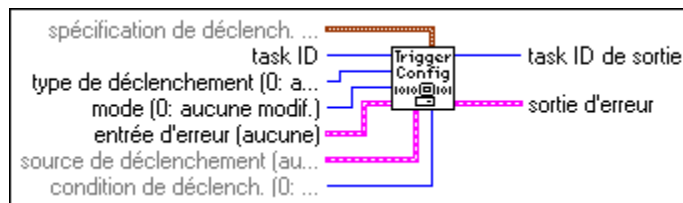
Ecrire une seule valeur numérique (Digital Single Write.vi)

Ecrit les données du **tableau de patterns** dans les voies numériques appartenant au groupe identifié par **taskID**.



Configurer un déclenchement numérique (Digital Trigger Config.vi)

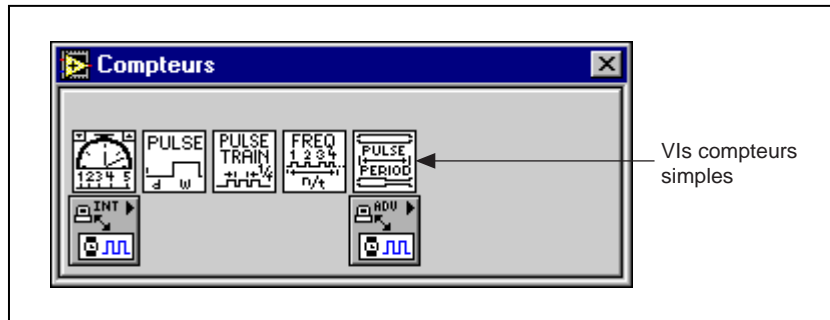
Configure la condition de déclenchement pour démarrer et/ou arrêter une opération de génération de modèle numérique. Ce VI n'est valable que si le paramètre **source du handshaking** du VI "Configurer une horloge numérique" (Digital Clock Config.vi) est réglé sur 1 ou 4 (génération de modèle externe ou interne avec horloge externe).



VIs Compteurs simples

Ce VI décrit les VIs qui exécutent des opérations de comptage simples. Vous pouvez exécuter ces VIs à partir de la face-avant ou les utiliser comme des sous-VIs dans des applications de base.

Vous pouvez accéder aux VIs Compteurs simples en choisissant **Fonctions»Acquisition de données»Compteurs**. Les VIs Compteurs simples se trouvent sur la rangée du haut de la palette **Compteurs**.



Ce chapitre décrit les VIs de haut niveau pour les compteurs de programmation des périphériques MIO, TIO et ceux dotés des circuits de compteur/timer DAQ-STC ou Am9513. Ces VIs appellent les VIs Compteurs intermédiaires pour générer une impulsion seule TTL avec retard, une série d'impulsions finie ou continue et pour mesurer la fréquence, la largeur d'impulsion ou la période d'un signal TTL.



Remarque

Ces VIs ne fonctionnent pas avec les périphériques des séries 1200 et Lab, DAQCard, et ceux dotés du circuit 8253/54. Utilisez la commande de niveau intermédiaire ICTR pour ces périphériques. Reportez-vous au chapitre 27, [VIs Compteurs intermédiaires](#), pour plus d'informations sur le VI "ICTR-Contrôler" (ICTR Control.vi).

Certains de ces VIs utilisent d'autres compteurs en plus de celui spécifié. Dans ce cas, un compteur logiquement adjacent est choisi, puis appelé **compteur+1** lorsqu'il est le compteur adjacent logiquement le plus haut et **compteur-1** lorsqu'il est le compteur adjacent logiquement le plus bas.

Dans le cas d'un périphérique doté du circuit Am9513, si le compteur est 1, alors **compteur+1** est compteur 2 et **compteur-1** est compteur 5.

Voir les VIs Compteurs adjacents décrits au chapitre 27, *VIs Compteurs intermédiaires*, pour plus d'informations.

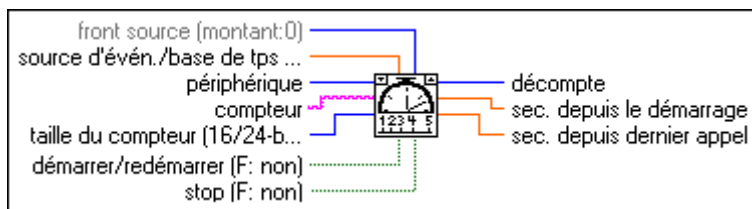
Pour des exemples sur la méthode d'utilisation des VIs Compteurs simples, consultez les bibliothèques d'exemples situées dans le répertoire `examples\daq\counter`.

Description des VIs Compteurs simples

Les VIs Compteurs simples suivants sont disponibles.

Compter des événements ou un temps (Count Events or Time.vi)

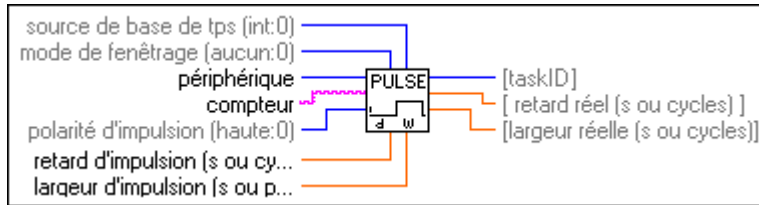
Configure un ou deux compteurs pour compter les événements externes ou du temps écoulé. Un événement externe est un signal de transition bas ou haut sur la broche SOURCE spécifiée du compteur.



Pour décompter des événements, réglez **source d'événements/base de temps** sur 0,0 et connectez le signal que vous voulez décompter sur la broche SOURCE du compteur. Pour décompter du temps, réglez cette commande sur la fréquence de la base de temps que vous voulez utiliser.

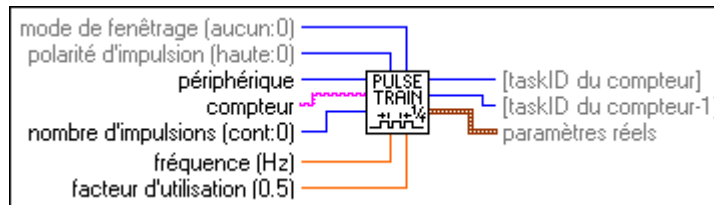
Générer une impulsion retardée (Generate Delayed Pulse.vi)

Configure et démarre un compteur pour générer une impulsion unique avec la largeur d'impulsion et le retard spécifiés sur la broche OUT du compteur. Une impulsion unique consiste en une phase de retard (phase 1), suivie d'une phase d'impulsion (phase 2) puis d'un retour à la phase 1. Si une base de temps interne est choisie, le VI sélectionne la base de temps avec la résolution la plus haute pour que le compteur réalise les caractéristiques désirées. Si un signal de base de temps externe est choisi, l'utilisateur indique le retard et la largeur en nombre de périodes de ce signal. Exécute le VI "Démarrer un compteur" (Counter Start.vi) avec le taskID de ce VI pour générer une autre impulsion. Eventuellement, vous pouvez fenêtrer ou déclencher l'impulsion par un signal sur la broche GATE du compteur.



Générer un train d'impulsions (Generate Pulse Train.vi)

Configure le compteur spécifié pour générer une série continue d'impulsions sur la broche OUT du compteur, ou pour générer une série d'impulsions de longueur finie en utilisant le compteur spécifié et un compteur adjacent. Le signal a la fréquence, le facteur d'utilisation et la polarité prescrits. Chaque cycle de la série d'impulsions consiste en une phase de retard (phase 1) suivie d'une phase d'impulsion (phase 2).



Ce VI n'utilise que le **compteur** spécifié pour générer une impulsion continue. Pour une impulsion de longueur finie, le VI utilise également **compteur-1** pour générer une impulsion à retard minimum pour le fenêtrage de **compteur**. Pour générer une autre série d'impulsions, exécutez le VI "Démarrer un compteur" (Counter Start.vi) intermédiaire avec les **taskID** fournis par ce VI. Pour arrêter une série d'impulsions continue, exécutez le VI intermédiaire "Arrêter le compteur" (Counter Stop.vi) ou exécutez ce compteur à nouveau pour générer une impulsion courte. Vous devez câbler extérieurement la broche OUT du **compteur-1** à la broche GATE du **compteur** pour une impulsion de longueur finie. Eventuellement, vous pouvez fenêtrer ou déclencher le début de la série par un signal sur la broche GATE du **compteur-1**.

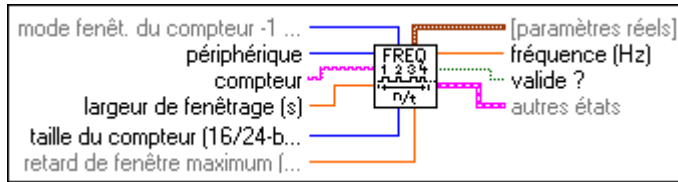


Remarque *Un train d'impulsions consiste en une série d'impulsions retardées, dans laquelle la phase 1 ou la première phase de chaque impulsion est l'état inactif de la sortie (bas pour une impulsion haute) et la phase 2 de la seconde phase est l'impulsion elle-même.*

Mesurer une fréquence (Measure Frequency.vi)

Mesure la fréquence d'un signal TTL sur la broche SOURCE du compteur spécifié en comptant les fronts positifs du signal pendant une période de temps spécifiée. En plus de cette connexion, vous devez câbler la broche GATE du compteur à la broche OUT du **compteur-1**. Ce VI est utile pour les signaux de fréquence relativement haute quand plusieurs périodes du

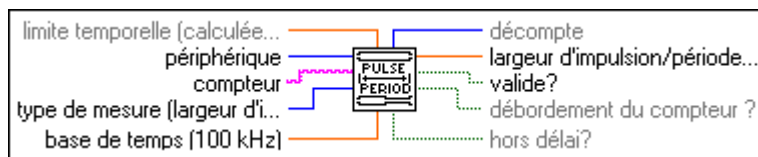
signal se produisent pendant la période de séquençement. Utilisez le VI “Mesurer une largeur d’impulsion ou une période” (Measure Pulse Width or Period.vi) pour les signaux de fréquence relativement basse. N’oubliez pas que période (s)= 1/fréquence (Hz).



Ce VI configure **compteur** et **compteur+1** (en option sur Am9513) spécifiés comme des compteurs destinés à décompter les fronts montants du signal sur la broche SOURCE du compteur. Le VI configure également **compteur-1** pour générer une impulsion à retard minimum pour fenêtrer le compteur d’événements, démarre le compteur d’événements puis le compteur de fenêtrage, attend la période de fenêtrage escomptée puis lit le compteur de fenêtrage jusqu’à ce que son état de sortie soit bas. Ensuite, le VI lit le compteur d’événements, calcule la fréquence du signal (**nombre d’événements/largeur d’impulsion de fenêtrage réelle**) et arrête les compteurs. Eventuellement, vous pouvez fenêtrer ou déclencher l’opération par un signal sur la broche GATE du **compteur-1**.

Mesurer une largeur d’impulsion ou une période (Measure Pulse Width or Period.vi)

Mesure la largeur d’impulsion (durée pendant laquelle un signal est haut ou bas) ou la période (temps écoulé entre les fronts descendants ou adjacents) d’un signal TTL connecté à la broche GATE du compteur. La méthode utilisée fenêtre une horloge de base de temps interne avec le signal mesuré. Ce VI est utile pour mesurer la période ou la fréquence (1/période) de signaux de fréquence relativement basse lorsque plusieurs périodes de base de temps se produisent pendant le fenêtrage. Utilisez le VI “Mesurer une fréquence” (Measure Frequency.vi) pour mesurer la période ou la fréquence de signaux de fréquence relativement haute.



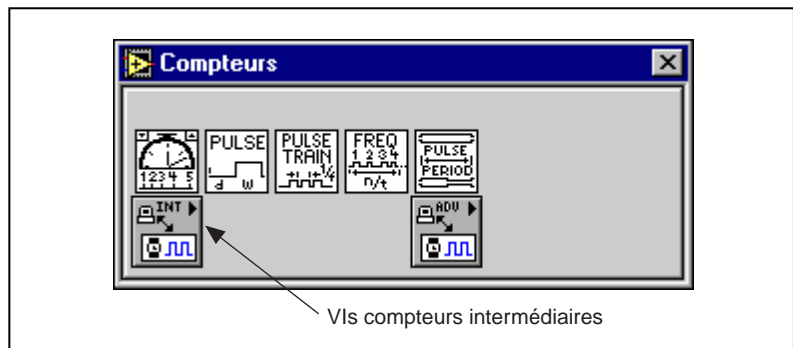
Ce VI s’exécute jusqu’à ce qu’une mesure valide, un **timeout** ou un débordement de **compteur** se produise. Une mesure valide existe lorsque **décompte** ≥ 4 sans débordement de compteur. Si un **débordement de compteur** se produit, diminuez la **base de temps**. Si vous démarrez une largeur d’impulsion pendant la phase que vous voulez mesurer, vous obtenez une mesure basse incorrecte. Assurez-vous donc que l’impulsion ne se produise pas avant le démarrage du **compteur**. Cette restriction ne s’applique pas aux mesures de périodes.

VIs Compteurs intermédiaires

Ce chapitre décrit les VIs Compteurs intermédiaires que vous pouvez utiliser sur les périphériques MIO, TIO et ceux dotés de circuits de compteur DAQ-STC ou Am9513. Ces VIs appellent les VIs Compteurs avancés pour configurer les compteurs propres aux opérations communes et pour démarrer, lire et arrêter les compteurs. Vous pouvez configurer ces VIs pour générer des impulsions uniques ou des séries d'impulsions continues, pour compter des événements et du temps écoulé, pour diviser un signal et pour mesurer une largeur d'impulsion ou une période. Les VIs Compteurs simples appellent les VIs Compteurs intermédiaires pour plusieurs opérations de génération d'impulsion, de décompte et de mesure.

Ce chapitre décrit également le VI "ICTR-Contrôle" (ICTR Control.vi) que vous pouvez utiliser avec les périphériques des séries 1200 et Lab, et les périphériques PC-LPM qui sont dotés du circuit compteur/timer.

Vous pouvez accéder aux VIs Compteurs intermédiaires en choisissant **Fonctions»Acquisition de données»Compteurs»Compteurs intermédiaires**. Les VIs Compteurs intermédiaires se trouvent sur la deuxième rangée de la palette **Compteurs**, comme illustré ci-dessous.



Gestion d'erreurs

Dans les VIs d'entrées/sorties intermédiaires de LabVIEW, la gestion d'erreur est simplifiée. Chaque VI de niveau intermédiaire comporte un cluster d'entrée **entrée d'erreur** et un cluster de sortie **sortie d'erreur**. Les clusters comprennent un booléen qui indique la présence ou non d'une erreur, le code d'erreur et le nom du VI qui a retourné l'erreur. Si **entrée d'erreur** indique une erreur, le VI transmet les informations au cluster **sortie d'erreur** et arrête l'exécution.

Quand vous utilisez n'importe quel VI Compteur intermédiaire dans une boucle While, vous devez arrêter la boucle si le booléen **état** dans le cluster **sortie d'erreur** indique Vrai (TRUE). Si vous câblez le cluster d'erreur sur le VI "Gestionnaire général d'erreurs" (General Error Handler.vi), le VI déchiffre l'information d'erreur et vous décrit l'erreur.

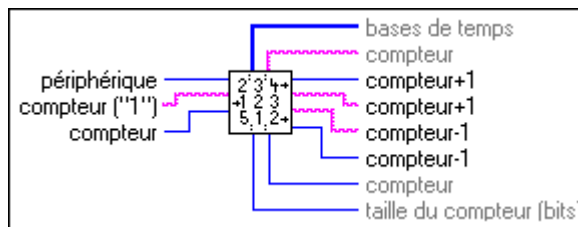
Le VI "Gestionnaire général d'erreurs" (General Error Handler.vi) se trouve dans la palette **Fonctions»Utilitaires** de LabVIEW. Pour plus d'informations sur ce VI, reportez-vous à votre *Manuel de l'utilisateur LabVIEW*.

Description des VIs Compteurs intermédiaires

Les VIs Compteurs intermédiaires disponibles sont les suivants.

Compteurs adjacents (Adjacent Counters.vi)

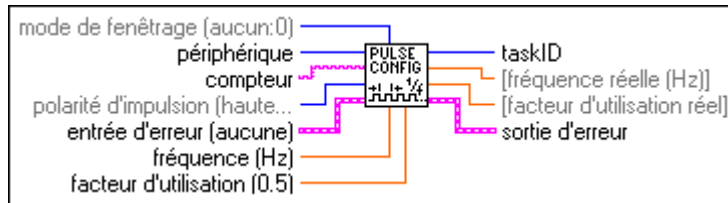
Identifie les compteurs directement adjacents à un compteur spécifié d'un périphérique MIO ou TIO. Il retourne également la taille du compteur (nombre de bits) et les bases de temps.



Les périphériques dotés du circuit Am9513 ont un ou deux ensembles de cinq compteurs 16 bits (1–5, 6–10) pouvant être connectés d'une façon circulaire. Par exemple, le compteur supérieur suivant le compteur 1 (appelé **compteur+1**) est 2 et le compteur immédiatement inférieur (appelé **compteur-1**) est 5.

Configurer un générateur continu d'impulsions (Continuous Pulse Generator Config.vi)

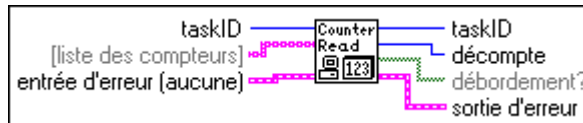
Configure un compteur pour générer une série d'impulsions TTL sur sa broche OUT.



Le signal est créé en décrémentant le compteur deux fois pour chaque impulsion, d'abord pour le retard de l'impulsion (phase 1) puis, pour l'impulsion elle-même (phase 2). Le VI sélectionne la base de temps à résolution la plus haute pour obtenir les caractéristiques désirées. Eventuellement, vous pouvez fenêtrer ou déclencher l'opération par un signal sur la broche GATE du compteur. Appelez le VI "Démarrer un compteur" (Counter Start.vi) pour démarrer la série d'impulsions ou pour la valider par un fenêtrage.

Lire un compteur (Counter Read.vi)

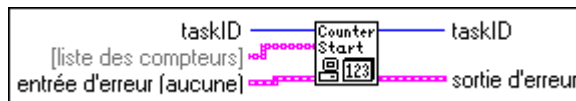
Lit le compteur ou les compteurs identifiés par **taskID**.



Le VI est conçu pour lire un compteur ou deux compteurs concaténés d'un circuit de compteur Am9513 ou pour lire le compteur d'un circuit DAQ-STC.

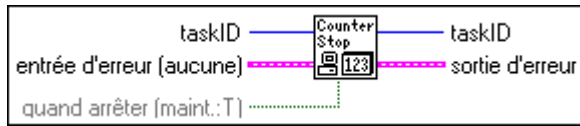
Démarrer un compteur (Counter Start.vi)

Démarre les compteurs identifiés par **taskID**.



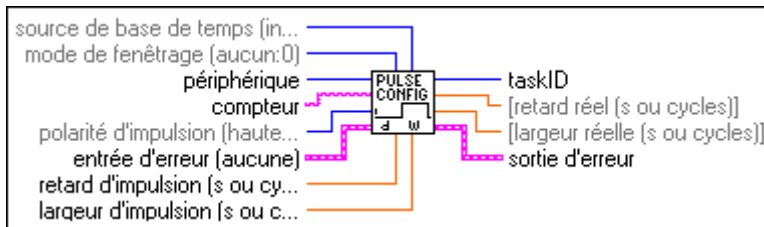
Arrêter un compteur (Counter Stop.vi)

Arrête une opération de décompte, immédiatement ou sous condition, sur une erreur d'entrée.



Configurer un générateur d'impulsions retardées (Delayed Pulse Generator Config.vi)

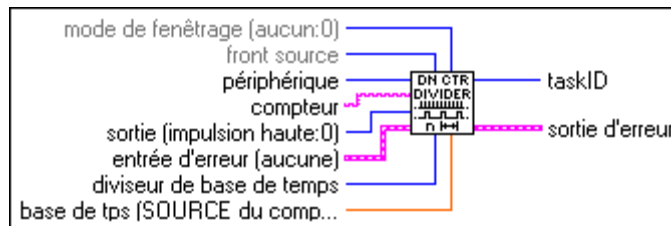
Configure un compteur pour générer une impulsion TTL à retard unique sur sa broche OUT.



Le signal est créé en décrémentant le **compteur** deux fois pour chaque impulsion, d'abord pour le retard de l'impulsion (phase 1) puis, pour l'impulsion elle-même (phase 2). Si une base de temps est choisie, le VI sélectionne la base de temps à résolution la plus haute pour que le **compteur** accomplisse les caractéristiques désirées. Si un signal de base de temps externe est choisi, l'utilisateur désigne le retard et la largeur en spécifiant le nombre de périodes de ce signal. Eventuellement, vous pouvez fenêtrer ou déclencher l'opération par un signal sur la broche GATE du compteur. Appelez le VI "Démarrer un compteur" (Counter Start.vi) pour démarrer l'impulsion ou la valider par un fenêtrage.

Configurer un compteur descendant ou diviseur (Down Counter or Divider Config.vi)

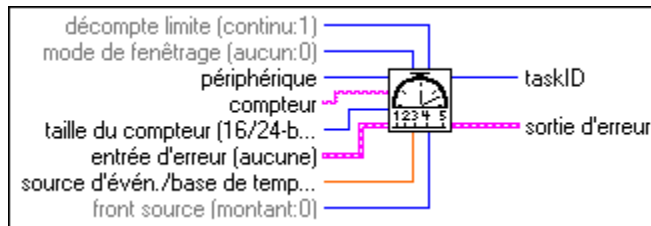
Configure le **compteur** spécifié pour décompter ou diviser un signal sur la broche SOURCE du compteur ou sur un signal de base de temps interne en utilisant une valeur de décompte appelée **diviseur de base de temps**. Il en résulte que le signal sur la broche OUT du compteur est égal à la fréquence du signal d'entrée divisée par le **diviseur de base de temps**.



Vous pouvez utiliser ce VI pour générer des séries d’impulsions finies en validant un générateur d’impulsions continues jusqu’à ce que le nombre d’impulsions désiré se soit produit. Vous pouvez également l’utiliser au lieu du VI “Configurer un générateur continu d’impulsions” (Continuous Pulse Generator Config.vi) pour générer une série de signaux de déclenchement ou de sonde.

Configurer un compteur de temps ou d’événement (Event or Time Counter Config.vi)

Configure un ou deux compteurs pour compter les fronts du signal sur la broche SOURCE du compteur spécifié ou le nombre de périodes d’un signal de base de temps interne spécifié.

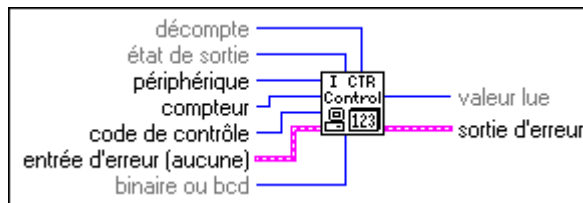


Lorsque la base de temps interne est utilisée, ce VI fonctionne comme la fonction “Compteur d’impulsion (ms)” mais il utilise un compteur matériel du périphérique DAQ avec résolution programmable. Eventuellement, vous pouvez fenêtrer ou déclencher une opération par un signal sur la broche GATE du compteur. Appelez le VI “Démarrer un compteur” (Counter Start) pour démarrer l’opération ou la valider par un fenêtrage.

ICTR-Contrôle (ICTR Control.vi)

Contrôle des compteurs sur des périphériques qui utilisent le circuit 8253/54, comprenant :

- Les périphériques des séries 1200 et Lab, DAQCard-500 et DAQCard-700
- Les périphériques (**Windows**) LPM et 516



En mode de configuration 0, comme illustré dans la figure 27-1, la sortie passe au niveau bas après l'opération de réglage de mode et le **compteur** commence à décompter pendant que l'entrée de fenêtrage est à l'état haut. La sortie passe au niveau haut lorsque le **compteur** atteint le TC (c'est-à-dire lorsque le compteur décroît jusqu'à 0) et reste au niveau haut jusqu'à ce que vous réglez le compteur sélectionné sur un mode différent.

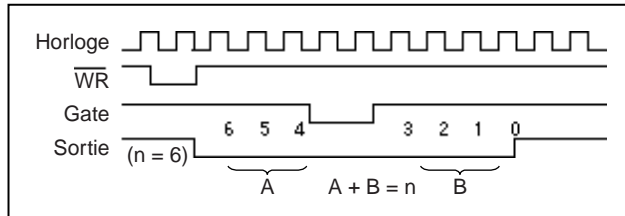


Figure 27-1. Mode de configuration dans "ICTR-Contrôler" (ICTR Control.vi)

En mode de configuration 1, comme illustré dans la figure 27-2, la sortie passe au niveau bas sur le **décompte** suivant le front montant de l'entrée de fenêtrage et passe au niveau haut sur TC.

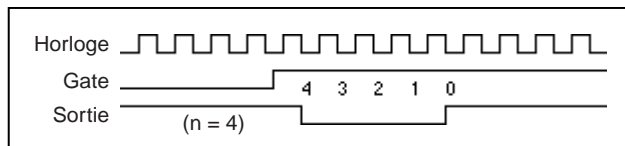


Figure 27-2. Mode de configuration 1 dans "ICTR-Contrôler" (ICTR Control.vi)

En mode de configuration 2, comme illustré dans la figure 27-3, la sortie passe au niveau bas pendant une période de l'entrée d'horloge. Le **décompte** indique la période entre les impulsions de sortie.

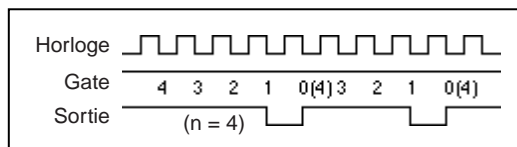


Figure 27-3. Mode de configuration 2 dans "ICTR-Contrôler" (ICTR Control.vi)

En mode de configuration 3, la sortie reste au niveau haut pendant la moitié des impulsions d'horloge **décompte** et passe au niveau bas pour l'autre moitié. Reportez-vous à la figure 27-4.



Remarque *Le décompte n'est possible que lorsque la valeur de fenêtrage reste au niveau haut.*

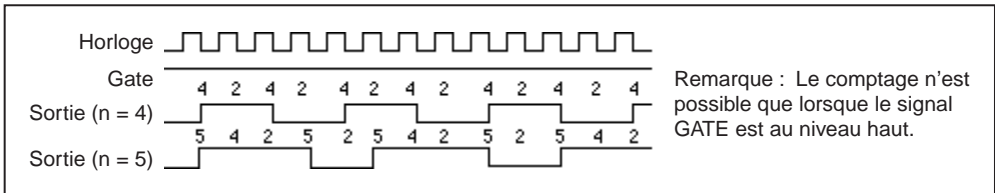


Figure 27-4. Mode de configuration 3 dans "ICTR-Contrôler" (ICTR Control.vi)

En mode de configuration 4, comme dans la Figure 27-5, la sortie est initialement au niveau haut et le **compteur** commence à décompter pendant que l'entrée de fenêtrage est au niveau haut. Sur TC, la sortie passe au niveau bas pour une impulsion d'horloge puis elle repasse au niveau haut.

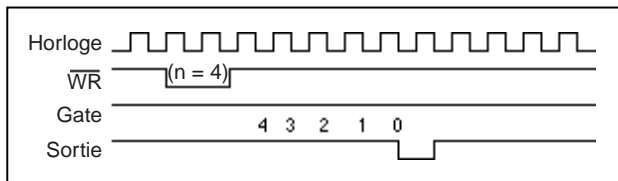


Figure 27-5. Mode de configuration 4 dans "ICTR-Contrôler" (ICTR Control.vi)

Le mode de configuration 5 est semblable au mode 4, sauf que l'entrée de fenêtrage déclenche le commencement du décompte. Voir la Figure 27-6 pour une illustration du mode 5.

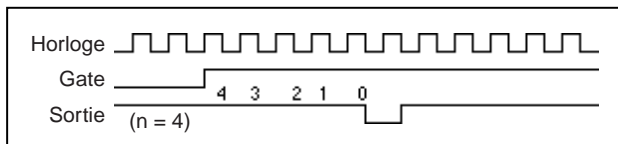
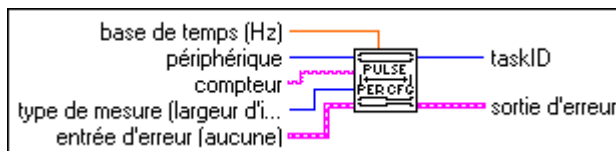


Figure 27-6. Mode de configuration 5 dans "ICTR-Contrôler" (ICTR Control.vi)

Voir la feuille de données Programmable Interval Timer 8253 dans le manuel de l'utilisateur de votre carte Lab pour des détails sur ces modes et leurs diagrammes de séquencement d'exécution associés.

Configurer une mesure de période ou une largeur d'impulsion (Pulse Width or Period Meas Config.vi)

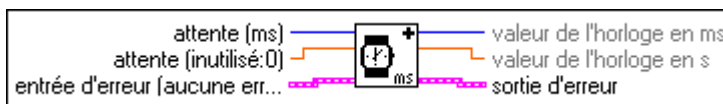
Configure le **compteur** spécifié pour mesurer la largeur d'impulsion ou la période d'un signal TTL connecté à sa broche GATE.



La mesure est prise en comptant le nombre de périodes de la base de temps spécifiée entre les événements de début et de fin. Pour mesurer avec précision la largeur d'impulsion, l'impulsion doit se produire après le démarrage du **compteur**. Appelez le VI "Démarrer un compteur" (Counter Start) pour démarrer l'opération. Vous pouvez également utiliser ce VI pour mesurer la fréquence des signaux de basse fréquence. Pour des mesures plus précises, utilisez une **base de temps** plus rapide.

Attendre+ (ms) (Wait+ (ms).vi)

Appelle la fonction "Attendre (ms)" uniquement en cas d'absence d'erreur en entrée.

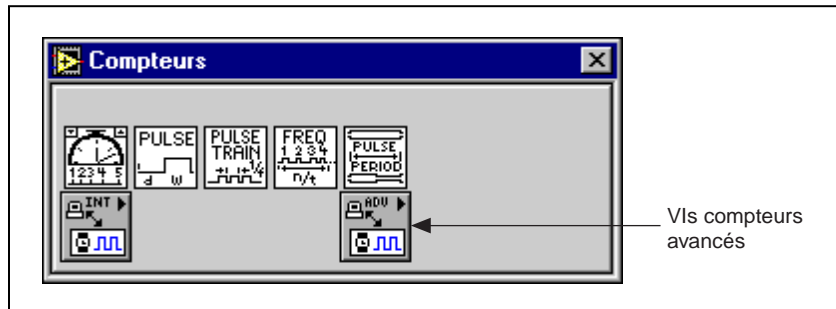


Ce VI est utile lorsque vous voulez attendre entre des appels à des sous-VIs d'E/S qui utilisent le mécanisme d'erreur d'E/S ; sans ce VI, vous avez besoin d'utiliser une structure Séquence pour commander l'ordre d'exécution.

Vis Compteurs avancés

Ce chapitre décrit les VIs qui configurent et contrôlent les compteurs matériels. Vous pouvez utiliser ces VIs pour générer des ondes carrées avec des facteurs d'utilisation variables, pour compter des événements ou du temps et pour mesurer des périodes et des fréquences.

Vous pouvez accéder à la palette **Compteurs avancés** en choisissant **Fonctions»Acquisition de données»Compteurs»Compteurs avancés**. L'icône que vous devez sélectionner pour accéder aux VIs Compteurs avancés se trouve sur la rangée du bas de la palette **Compteurs**, comme illustré ci-dessous.



Remarque *N'utilisez que les entrées dont vous avez besoin sur chaque VI quand vous travaillez avec des acquisitions de données. Ne câblez pas les entrées restantes, LabVIEW les règle sur des valeurs par défaut. Dans la fenêtre Aide, les terminaux les plus importants sont étiquetés en caractères gras, et les terminaux les moins utilisés sont entre crochets. Les valeurs par défaut sont données entre parenthèses.*

Les types de circuits de compteurs que votre périphérique doit avoir pour fonctionner avec votre version de LabVIEW, sont listés ci-dessous :

- Circuit de compteurs DAQ-STC
- Circuit de compteurs Am9513
- Circuit de compteurs 8253/54

Le VI "ICTR-Contrôler" (ICTRControl.vi) est le seul VI fonctionnant avec des périphériques contenant le circuit de compteurs 8253/54.

Reportez-vous au tableau 28-1 pour les circuits de compteurs utilisés avec les différents périphériques.

Tableau 28-1. Circuits de compteurs et leurs périphériques DAQ disponibles

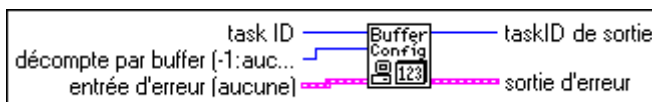
Circuit de compteurs	Périphérique DAQ
Am9513	AT-MIO-16, AT-MIO-16D, AT-MIO-16F-5, AT-MIO-16X, AT-MIO-64F-5, PC-TIO-10, tous les périphériques AO-2DC, EISA-A2000, NB-MIO-16, NB-MIO-16X, NB-DMA-8-G, NB-DMA2800, NB-TIO-10, NB-A2000
DAQ-STC	Tous les périphériques Série E, périphériques 5102
8253/54	Tous les périphériques des séries Lab et 1200, DAQCard-500, DAQCard-700, périphériques LPM, périphériques 516

Description des VIs Compteurs avancés

Les VIs Compteurs avancés disponibles sont les suivants.

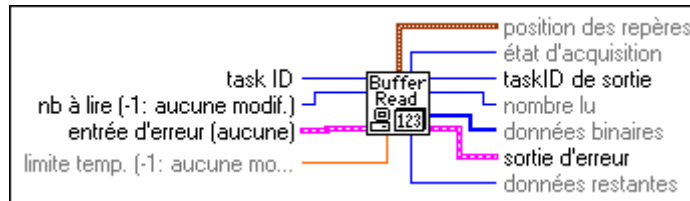
CTR-Configurer un buffer (CTR Buffer Config.vi)

Alloue de la mémoire pour que LabVIEW puisse stocker les données de compteur. Le VI “CTR-Configurer un buffer” (CTR Buffer Config.vi) configure également le groupe spécifié pour exécuter des opérations de compteurs bufférisées au lieu d’opérations point par point classiques.



CTR-Lire un buffer (CTR Buffer Read.vi)

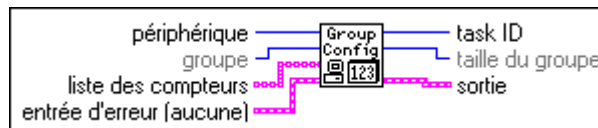
Retourne des données du buffer alloué par “CTR-Configurer un buffer” (CTR Buffer Config.vi).



Remarque Une lecture du buffer de compteur par incrémentation est supportée. Toutefois, une utilisation circulaire du buffer n’est pas implémentée. Vous devez donc configurer un buffer fini. Vous pouvez lire un buffer fini au fur et à mesure de son remplissage.

CTR-Configurer un groupe (CTR Group Config.vi)

Rassemble un ou plusieurs compteurs dans un groupe. Vous pouvez utiliser des groupes de compteurs contenant plusieurs compteurs pour démarrer, arrêter ou lire plusieurs compteurs simultanément. Les périphériques DAQ-STC ne supportent pas actuellement plusieurs compteurs dans un même groupe.



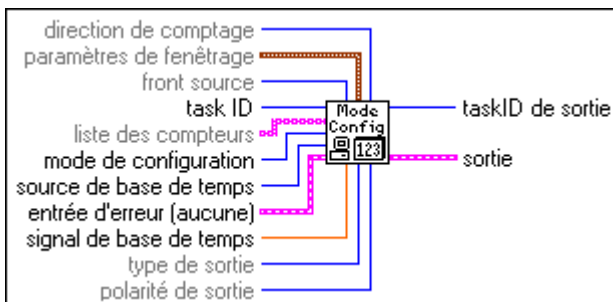
Le tableau 28-2 contient des numéros de compteurs valides pour des périphériques supportés par ce VI.

Tableau 28-2. Numéros de compteurs pour les périphériques “CTR-Configurer un groupe” (CTR Group Config.vi)

Type de périphérique	Numéros valables
Périphériques DAQ-STC	0 et 1
Périphériques Am9513 MIO	1, 2 et 5
NB-DMA-8-G, NB-DMA2800	1 à 5
PC-TIO-10, NB-TIO-10	1 à 10
EISA-A2000, NB-A2000	2

CTR-Configurer un mode (CTR Mode Config.vi)

Configure un ou plusieurs compteurs pour une opération de compteurs déterminée et sélectionne le signal source, le mode de fenêtrage et le comportement de sortie sur décompte terminal (TC).



Ce VI ne démarre pas les compteurs. Utilisez le VI “CTR-Contrôler” (CTR Control.vi) avec **code de contrôle** égal à 1 (Démarrer) pour démarrer les compteurs. Si vous utilisez un compteur pour la génération d’impulsion, il n’est pas nécessaire d’appeler ce VI à moins que vous ne vouliez changer le **mode de fenêtrage** ou le comportement des sorties.

Les modes 3, 4 et 6 peuvent être utilisés avec ou sans comptage bufférisé. Le mode 7 doit être utilisé avec le comptage bufférisé. Dans le cas de comptage bufférisé, appelez le VI “CTR-Configurer un buffer” (CTR Buffer Config.vi), avant ou après le VI “CTR-Configurer un mode” (CTR Mode Config.vi) et après le VI “CTR-Contrôler” (CTR Control.vi) pour démarrer l’opération, puis appelez le VI “CTR-Lire un buffer” (CTR Buffer Read.vi) pour lire les valeurs de comptage bufférisé. Dans le cas d’opérations bufférisées ou non bufférisées, appelez le VI “CTR-Contrôler” (CTR Control.vi) pour lire la valeur de comptage (non bufférisée) la plus récemment acquise.

Sauf indication contraire, les figures suivantes contiennent des valeurs de compteurs et de séquençement d’exécution pour des opérations dans lesquelles le **mode de fenêtrage** est réglé sur un niveau haut ou un front montant et le **front source** est réglé sur front montant.

Utilisez le mode 1 pour restaurer les réglages par défaut de tous les paramètres du VI “CTR-Configurer un mode” (CTR Mode Config.vi). Ce mode se substitue à tous les réglages de paramètres conflictuels.

Utilisez le mode 2 pour compter des transitions du signal sélectionné et arrêter au premier TC. Le bit d’état de débordement est réglé sur TC. Utilisez le VI “CTR-Contrôler” pour lire l’état de débordement. Ce mode n’est disponible que sur les périphériques Am9513. Le comptage en mode 2 est non bufférisé. La figure 28-1 illustre les valeurs de comptage que vous pourriez lire dans ce mode en utilisant deux réglages de **mode de fenêtrage** (fenêtrage sur niveau haut et fenêtrage sur front montant).

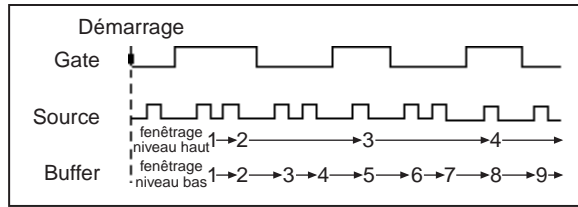


Figure 28-1. Modes 2 et 3 de comptage non bufférisés

Utilisez le mode 3 pour compter des transitions du signal sélectionné de façon continue, jusqu’au TC puis en continuant. La figure 28-1 présente le comptage en mode 3 non bufférisé. La figure 28-2 illustre une opération en mode 3 bufférisé avec fenêtrage sur front montant. Cette opération bufférisée n’est disponible que sur des périphériques DAQ-STC. Dans le cas d’une opération en mode 3 bufférisé, LabVIEW stocke la valeur de comptage courante dans le buffer sur chaque front sélectionné du signal source.

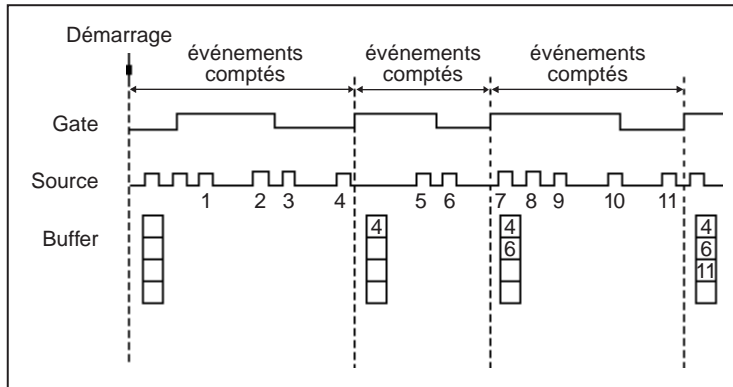


Figure 28-2. Mode 3 de comptage bufférisé

Utilisez le mode 4 avec fenêtrage sur niveau pour mesurer la largeur d’impulsion et le fenêtrage sur front pour mesurer la période du signal de fenêtrage sélectionné.



Remarque Dans le cas des descriptions de mesures de largeur d’impulsions suivantes (modes 4, 6 et 7), une impulsion haute est simplement définie comme la phase de niveau haut d’un signal quand le mode de fenêtrage est réglé sur fenêtrage sur niveau haut. Cette définition diffère de celle d’une impulsion haute utilisant la génération d’impulsion (mode 5), et qui consiste en une phase de retard de niveau bas suivie d’une phase d’impulsion de niveau haut. (Les impulsions de niveau bas sont définies de la même manière en permutant les mots haut et bas.)

Pour mesurer la largeur d’impulsion, réglez le **mode de fenêtrage** sur niveau haut ou bas. La figure 28-3 présente des mesures de largeur d’impulsion en mode 4 non bufférisé. Vous pouvez démarrer un compteur Am9513 à tout moment pour qu’il mesure des impulsions jusqu’à ce que vous l’arrêtiez. Si vous le démarrez au milieu de l’impulsion que vous voulez mesurer (par exemple, pendant une impulsion haute dans un fenêtrage sur niveau haut), LabVIEW retourne un compte court pour cette mesure. Vous ne devez démarrer un compteur DAQ-STC que lorsque le signal est dans la polarité opposée au niveau de fenêtrage sélectionné (par exemple, un fenêtrage sur niveau bas pour fenêtrage sur niveau haut). Autrement, le VI retourne le numéro d’erreur –10890. En cas de comptage non bufférisé, le DAQ-STC arrête le comptage après une mesure. Le mode 5 configure l’impulsion pour la génération d’impulsion. Utilisez le VI “CTR-Configurer une impulsion” (CTR Pulse Config.vi) pour spécifier l’impulsion que vous voulez générer.

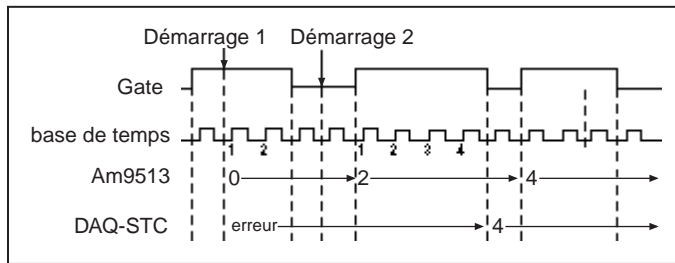


Figure 28-3. Mesure d’une largeur d’impulsion haute en mode 4 non bufférisé

La figure 28-4 présente la mesure de largeur d’impulsion en mode 4 bufférisé qui n’est disponible que sur les périphériques DAQ-STC. La valeur mesurée est stockée dans le buffer à la fin de chaque impulsion. Voir le mode 6 pour connaître une autre méthode de mesure de la largeur d’impulsion avec un périphérique DAQ-STC.

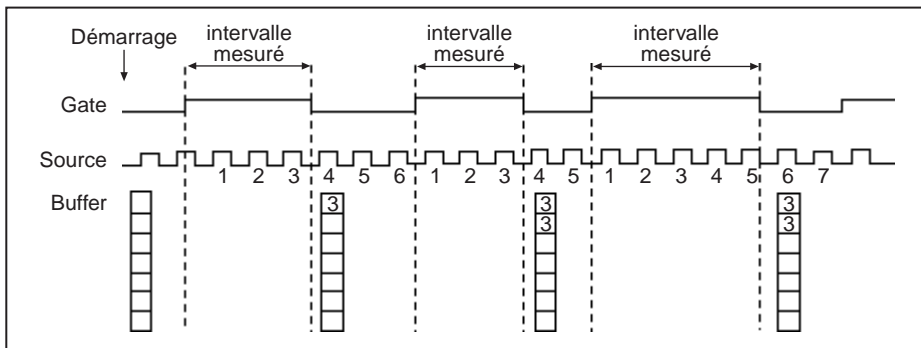


Figure 28-4. Mesure d’une largeur d’impulsion de front montant en mode 4 bufférisé

Pour mesurer des périodes, réglez le **mode de fenêtrage** sur front montant ou descendant. La figure 28-5 présente la mesure d'une largeur d'impulsion en mode 4 bufférisé.

Vous pouvez démarrer un compteur Am9513 ou bien un compteur DAQ-STC à tout moment. Le compteur commence le comptage au début de la période suivante. Le compteur Am9513 mesure des périodes de façon continue. En cas de comptage bufférisé, le DAQ-STC arrête le comptage après une mesure.

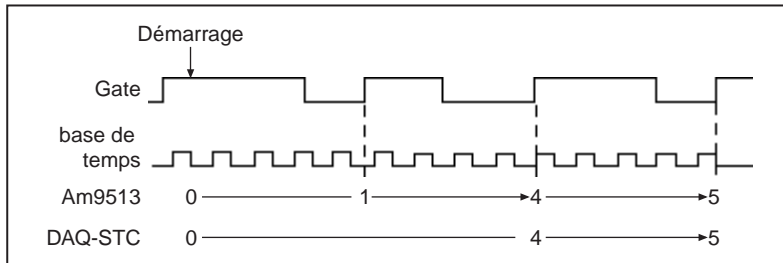


Figure 28-5. Mesure d'une période sur front montant en mode 4 non bufférisé

La figure 28-6 présente une mesure de période en mode 4 bufférisé qui n'est disponible que sur les périphériques DAQ-STC. La valeur mesurée est stockée dans le buffer à la fin de chaque période.

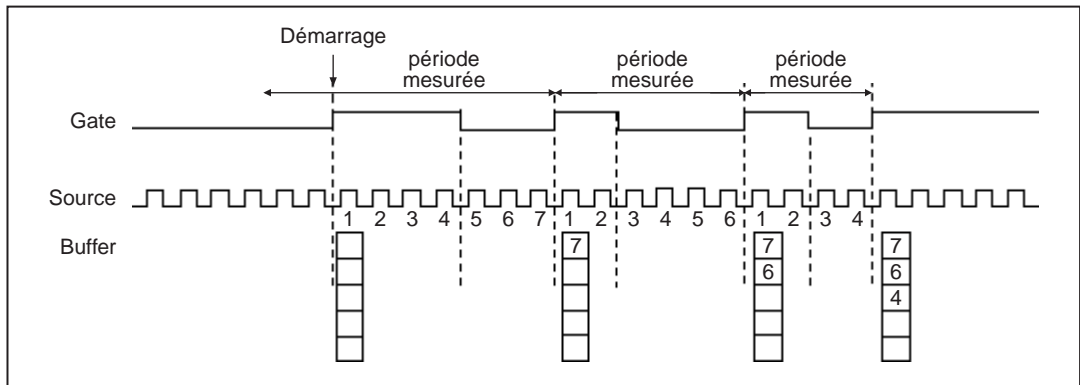


Figure 28-6. Mesure d'une largeur d'impulsion sur front montant en mode 4 bufférisé

Utilisez le mode 5 pour configurer la génération d'impulsion lorsque vous devez également configurer le **mode de fenêtrage**, **type de sortie** ou la **polarité de sortie** avec des valeurs implicites. Autrement, évitez d'appeler le VI "CTR-Configurer un mode" (CTR Mode Config.vi) et n'utilisez que le VI "CTR-Configurer une impulsion" (CTR Pulse Config.vi) pour la génération d'impulsion. Voir le VI "CTR-Configurer une impulsion" (CTR Pulse Config.vi) pour des informations supplémentaires concernant cette opération.

Utilisez le mode 6 avec fenêtrage sur niveau pour mesurer la largeur d'impulsion du signal sélectionné. Ce mode n'est disponible que sur des périphériques DAQ-STC. Le mode 6 diffère du mode 4 car la mesure d'une impulsion haute (basse) ne commence pas avant le premier front descendant (montant) du signal après que vous ayez démarré le compteur. Si vous utilisez un comptage non bufférisé, le compteur continue de mesurer des impulsions jusqu'à ce que vous appelez le VI "CTR-Contrôler" (CTR Control.vi) pour lire la valeur la plus récemment mesurée et, à ce moment-là, le compteur s'arrête. Le comptage en mode 6 non bufférisé est illustré dans la figure 28-7.

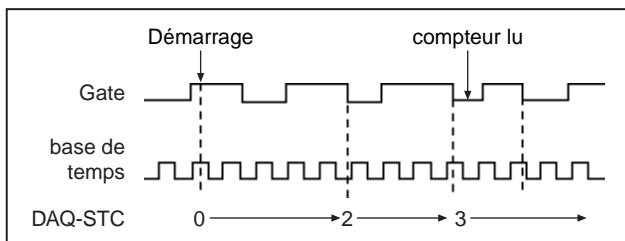


Figure 28-7. Mesure d'une largeur d'impulsion haute en mode 6 non bufférisé

Avec le comptage en mode 6 bufférisé, la valeur mesurée est stockée dans le buffer à la fin de chaque impulsion, comme illustré dans la figure 28-8. Appelez le VI "CTR-Lire un buffer" (CTR Buffer Read.vi) pour lire les valeurs.

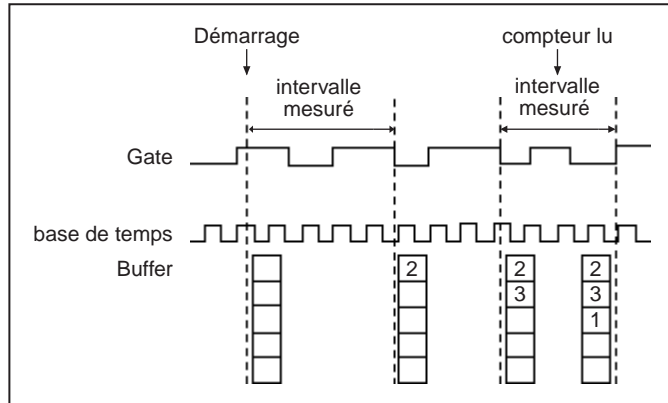


Figure 28-8. Mesure d'une largeur d'impulsion haute en mode 6 bufférisé (Décompte du front montant de la source)

Utilisez le mode 7 pour mesurer chaque phase du signal sélectionné en utilisant le comptage bufférisé. Ce mode n'est disponible que sur des périphériques DAQ-STC. La valeur de comptage est stockée dans le buffer lors de chaque transition de niveau bas vers haut et haut vers bas. Utilisez le VI "CTR-Lire un buffer" (CTR Buffer Read.vi) pour lire les valeurs. Pour mesurer des périodes avec ce mode, additionnez des paires successives de signaux. Pour mesurer des phases, utilisez une valeur sur deux. LabVIEW ignore la valeur du **mode de fenêtrage** en mode 7, ce qui signifie que vous ne pouvez pas savoir si la première mesure commence sur un front montant ou descendant.

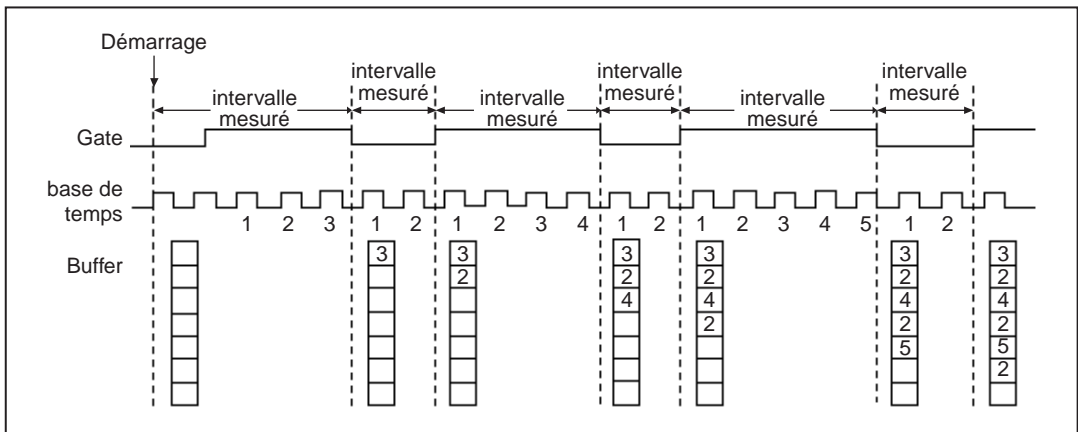


Figure 28-9. Mesure d'une semi-période en mode 7 bufférisé

Le tableau 28-3 présente les valeurs valides et les paramètres par défaut du **signal de base de temps**. Une valeur de -1 indique à LabVIEW d'utiliser les paramètres par défaut. Le terme compteur du tableau fait référence au compteur configuré. Dans le cas de plusieurs compteurs, LabVIEW configure chaque compteur successivement.

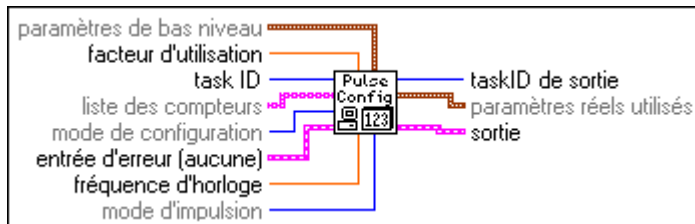
Reportez-vous au tableau 28-3 pour déterminer le compteur suivant ou précédent.

Tableau 28-3. Compteurs adjacents

Type de périphérique	Compteur inférieur suivant	Compteur	Compteur supérieur suivant
Am9513	5	1	2
	1	2	3
	2	3	4
	3	4	5
	4	5	1
	10	6	7
	6	7	8
	7	8	9
	8	9	10
	9	10	6
DAQ-STC	1	0	1
	0	1	0

CTR-Configurer une impulsion (CTR Pulse Config.vi)

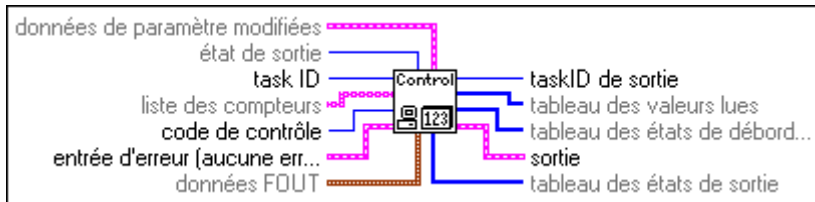
Spécifie les paramètres pour la génération d'impulsion. Ce VI configure les compteurs mais ne les démarre pas. Il utilise le VI "CTR-Contrôler" (CTR Control.vi) avec le code de commande égal à 1 (Démarrer) pour produire l'impulsion.



Utilisez ce VI pour spécifier les caractéristiques de vos impulsions. Vous pouvez également utiliser le VI “CTR-Configurer un mode” (CTR Mode Config.vi) pour régler les modes de fenêtrage, la polarité et le type de sortie que vous désirez. Utilisez le “CTR-Configurer une impulsion” (CTR Pulse Config.vi) pour spécifier la **source de base de temps** et le **signal de base de temps** pour la génération d’impulsion, parce que LabVIEW ignore ces valeurs spécifiées dans le VI “CTR-Configurer un mode” (CTR Mode Config.vi).

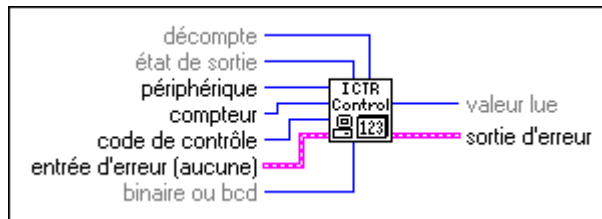
CTR-Contrôler (CTR Control.vi)

Contrôle et lit des groupes de compteurs. Les opérations de contrôle comprennent le démarrage, l’arrêt et le réglage de l’état de la sortie.



ICTR-Contrôler (ICTRControl.vi)

Contrôle des compteurs sur des périphériques qui utilisent le circuit 8253 (cartes des séries 1200 et Lab, périphériques 516-PC-LPM-16, DAQCard-500 et DAQCard 700).



Vis d'étalonnage et de configuration

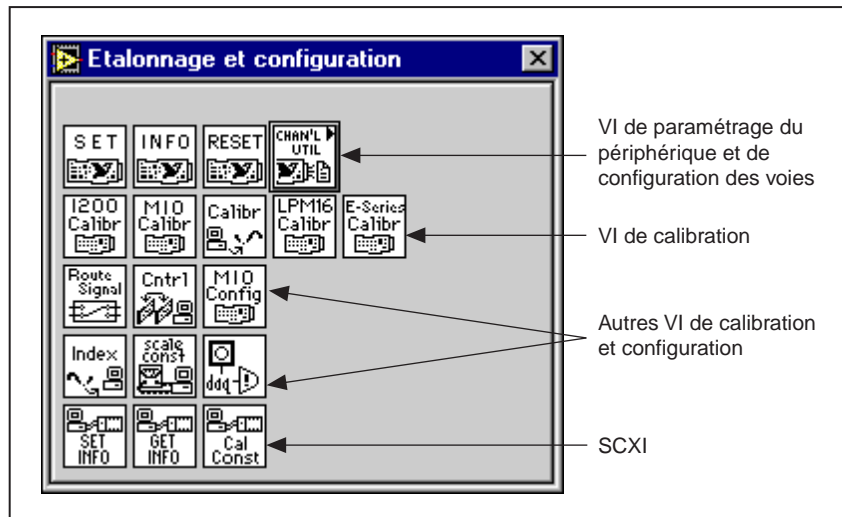
Ce chapitre décrit les VIs qui étalonnent des périphériques spécifiques et retournent des informations de configuration.

Ce chapitre comprend également un VI de contrôle du bus RTSI, qui est un bus de déclenchement et de séquençement que vous pouvez utiliser pour synchroniser, séquençer et déclencher plusieurs périphériques DAQ.

(Windows) Vous pouvez utiliser un VI pour configurer des occurrences d'événements d'acquisition de données.

Vous pouvez étalonner certains périphériques DAQ avec les VIs spécifiques aux périphériques, mais cela n'est pas toujours nécessaire dans la mesure où National Instruments étalonne tous ses périphériques en usine.

Vous pouvez accéder aux VIs d'étalonnage et de configuration en choisissant **Fonctions»Acquisition de données»Etalonnage et Configuration** comme illustré ci-dessous.

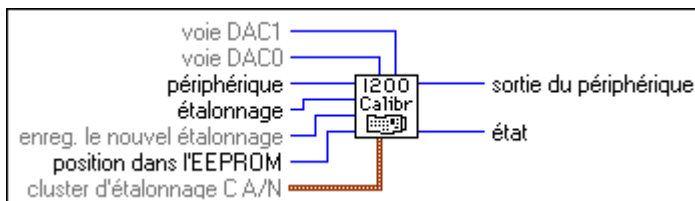


Description des VIs d'étalonnage et de configuration

Les VIs d'étalonnage et de configuration disponibles sont les suivants.

1200-Etalonner (1200 Calibrate.vi)

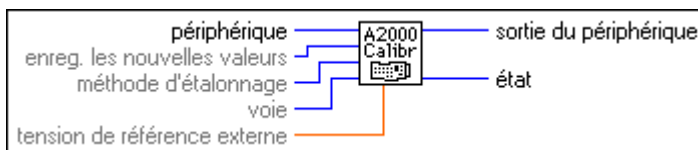
Ce VI étalonne les valeurs du gain et d'offset pour les C A/N et les C N/A des périphériques de série 1200 (par exemple, DAQPad-1200, DAQCard-1200, etc.).



Vous pouvez exécuter un nouvel étalonnage (et éventuellement, sauvegarder des constantes dans l'une des quatre zones utilisateur de l'EEPROM intégrée) ou charger un ensemble de constantes d'étalonnage existantes en les copiant à partir de leur position de stockage dans l'EEPROM intégrée. LabVIEW charge automatiquement les constantes d'étalonnage dans la zone de chargement de l'EEPROM intégrée quand LabVIEW est lancé ou quand vous réinitialisez le périphérique. La zone de chargement de l'EEPROM contient par défaut une copie des constantes d'étalonnage dans la zone d'usine.

A2000-Etalonner (A2000 Calibrate.vi)

Étalonne les valeurs du gain et d'offset des périphériques NB-A2000 ou EISA-A2000 A/N et restaure leurs valeurs d'origine réglées en usine.



Vous pouvez étalonner votre NB-A2000 ou EISA-A2000 pour ajuster les données des quatre voies de sorties analogiques. LabVIEW charge automatiquement les valeurs d'étalonnage stockées lorsqu'il procède au lancement ou quand vous réinitialisez votre NB-A2000 ou l'EISA-A2000.



Mise en garde

Ce VI n'est pas supporté sur des versions NI-DAQ ultérieures à la version 4.9.0 et a été supprimé de la palette d'étalonnage et de configuration. Ce VI est toujours inclus dans la bibliothèque de VIs DAQ pour la compatibilité seulement, c'est la raison pour laquelle ce VI retournera le message : `deviceSupportError`, si vous utilisez la version 5.0 de NI-DAQ ou une

version supérieure. Si vous souhaitez utiliser ce VI, veuillez réinstaller la version 4.9.0 de NI-DAQ ou une version antérieure.



Mise en garde *Reportez-vous à l'Annexe B, Capacités matérielles DAQ, pour plus d'informations sur le périphérique DAQ NB-A2000 ou EISA-A2000.*



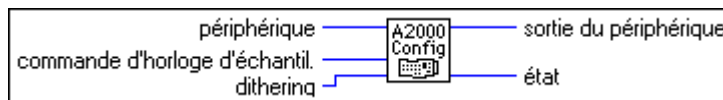
Mise en garde *Lisez le chapitre portant sur l'étalonnage dans le Manuel de l'utilisateur de NB-A2000 ou EISA-A2000 avant d'utiliser le VI "A2000-Etalonner" (A2000 Calibrate.vi).*

Si vous réglez le paramètre **enregistrer les nouvelles valeurs** sur 1, ce VI stocke alors les valeurs d'étalonnage de gain et d'offset dans une EEPROM sur le périphérique NB-A2000 ou EISA-A2000, qui ne perd pas ses données même si le périphérique n'est plus alimenté. LabVIEW lit ces valeurs contenues dans l'EEPROM et les charge dans le NB-A2000 ou l'EISA-A2000. Vous pouvez choisir de remplacer les copies permanentes des valeurs de gain et d'offset de l'EEPROM et utiliser les nouvelles valeurs jusqu'à l'étalonnage suivant, même si vous réinitialisez le périphérique. Vous pouvez également choisir de ne pas remplacer les valeurs de l'EEPROM mais d'utiliser les nouvelles valeurs jusqu'à l'initialisation ou l'étalonnage suivant.

Par exemple, si vous obtenez continuellement des données inexactes d'une ou de plusieurs voies d'entrée après avoir réinitialisé le périphérique, vous pouvez calibrer et sauvegarder les nouvelles valeurs de gain et d'offset comme des copies permanentes dans l'EEPROM. Cependant, si les résultats d'acquisition sont exacts après l'initialisation mais dérivent après quelques heures de fonctionnement du périphérique, quand la température monte, vous pouvez calibrer le périphérique à cette température de fonctionnement et garder les valeurs de l'EEPROM actuelles pour les utiliser après l'initialisation suivante.

A2000-Configurer (A2000 Config.vi)

Configure le dithering pour commander la ligne SAMPCLK* pour le NB-A2000 ou l'EISA-A2000.



Mise en garde *Ce VI n'est pas supporté sur des versions NI-DAQ ultérieures à la version 4.9.0 et a été supprimé de la palette d'étalonnage et de configuration. Ce VI est toujours inclus dans la bibliothèque de VIs DAQ pour la compatibilité seulement, c'est la raison pour laquelle ce VI retournera le message : `deviceSupportError`, si vous utilisez la version 5.0 de NI-DAQ ou une version ultérieure. Si vous souhaitez utiliser ce VI, veuillez réinstaller la version 4.9.0 de NI-DAQ ou une version antérieure.*

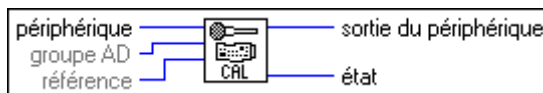
Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour plus d'informations sur le périphérique NB-A2000 ou EISA-A2000 DAQ.

Après le démarrage du système, LabVIEW configure le NB-A2000 ou l'EISA-A2000 comme suit :

- **signal d'horloge d'échantillonnage** = 0 : le signal d'horloge d'échantillonnage ne commande pas de ligne SAMPCLK*.
- **dithering** = 0 : dithering désactivé.

A2100-Etalonner (A2100 Calibrate.vi)

Sélectionne la référence d'étalonnage désirée et exécute un cycle d'étalonnage d'offset sur les C A/N du NB-A2100 ou du NB-A2150.

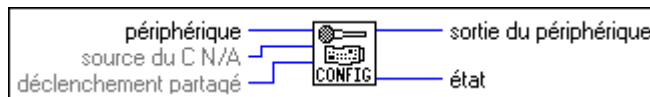


Mise en garde *Ce VI n'est pas supporté sur des versions NI-DAQ ultérieures à la version 4.9.0 et a été supprimé de la palette d'étalonnage et de configuration. Ce VI est toujours inclus dans la bibliothèque de VIs DAQ pour la compatibilité seulement, c'est la raison pour laquelle ce VI retournera le message : `deviceSupportError`, si vous utilisez la version 5.0 de NI-DAQ ou une version ultérieure. Si vous souhaitez utiliser ce VI, veuillez réinstaller la version 4.9.0 de NI-DAQ ou une version antérieure.*

Le logiciel driver NI-DAQ étalonne les deux voies A/N en utilisant la masse des sorties analogiques comme référence de chaque voie lorsque vous allumez l'ordinateur.

A2100-Configurer (A2100 Config.vi)

Sélectionne la source du signal utilisé pour fournir des données aux C N/A et vous laisse configurer le déclenchement numérique externe devant être partagé par les opérations d'acquisition de données et de génération de signaux sur le NB-A2100.



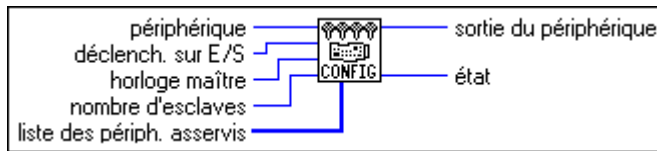
Mise en garde *Ce VI n'est pas supporté sur des versions NI-DAQ ultérieures à la version 4.9.0 et a été supprimé de la palette d'étalonnage et de configuration. Ce VI est toujours inclus dans la bibliothèque de VIs DAQ pour la compatibilité seulement, c'est la raison pour laquelle ce VI retournera le message : `deviceSupportError`, si vous utilisez la version 5.0 de NI-DAQ ou une*

version ultérieure. Si vous souhaitez utiliser ce VI, veuillez réinstaller la version 4.9.0 de NI-DAQ ou une version antérieure.

Si LabVIEW acquiert plusieurs cadres d'acquisition de données et génère plusieurs cycles de formes d'onde avec un déclenchement demandé au début de chaque cycle, la reconnaissance du déclenchement externe effectue une synchronisation pour que chaque déclenchement initie simultanément l'acquisition du cadre de données suivant, tout en générant la sortie du cycle de forme d'onde suivant.

A2150-Configurer (A2150 Config.vi)

Décide si LabVIEW doit ou ne doit pas diriger un déclenchement généré de façon interne sur le connecteur d'E/S du NB-A2150. Ce VI détermine également si LabVIEW doit diriger le signal d'horloge d'échantillonnage du NB-A2150 via le bus RTSI vers d'autres périphériques pour une acquisition de données synchronisée de périphériques multiples.




Mise en garde

Ce VI n'est pas supporté sur des versions NI-DAQ ultérieures à la version 4.9.0 et a été supprimé de la palette d'étalonnage et de configuration. Ce VI est toujours inclus dans la bibliothèque de VIs DAQ pour la compatibilité seulement, c'est la raison pour laquelle ce VI retournera le message : `deviceSupportError`, si vous utilisez la version 5.0 de NI-DAQ ou une version ultérieure. Si vous souhaitez utiliser ce VI, veuillez réinstaller la version 4.9.0 de NI-DAQ ou une version antérieure.

Activez le **lecteur de déclenchement e/s** seulement si vous avez exécuté le VI "RTSI-Contrôler" (RTSI Control.vi) pour recevoir le signal RTSITRIG* via le bus RTSI, ou si vous avez activé le déclenchement de niveau analogique à l'aide du VI "AI-Configurer un déclenchement" (AI Trigger Config.vi). Dans ces cas-là, vous pouvez suivre le signal envoyé à l'ensemble des circuits de déclenchement A/N sur la ligne EXTTRIG* du connecteur d'E/S après avoir démarré l'acquisition. Un front descendant du signal déclenche l'acquisition des données.

Le périphérique NB-A2150 utilise des signaux via le bus RTSI pour la synchronisation d'horloge d'échantillonnage entre deux ou plusieurs périphériques NB-A2150. L'ensemble de circuits de synchronisation d'horloge d'échantillonnage permet l'échantillonnage simultané sur plus de quatre voies en utilisant des périphériques NB-A2150 supplémentaires. Si **horloge maître** est égal à 1, **liste d'esclaves** doit contenir la liste des périphériques qui acceptent l'horloge d'échantillonnage du **périphérique**. Une fois que vous avez exécuté "A2150-Configurer" (A2150 Config.vi) avec **horloge maître** égale à 1 et **nombre d'esclaves**

supérieur à 0, vous ne pouvez pas utiliser le VI "AI-Configurer une horloge" (AI Clock Config.vi) pour régler la séquence de balayage pour des périphériques de **liste d'esclaves** jusqu'à ce que vous exécutiez à nouveau le VI "A2150-Configurer" (A2150 Config.vi) sur **périphérique** avec **horloge maître** égale à 1 et **nombre d'esclaves** égal à 0.

 **Remarque** *L'exécution de "A2150-Configurer" (A2150 Config.vi) avec l'horloge maître égale à 1 et le nombre d'esclaves égal à 0, reconfigure les périphériques se trouvant précédemment dans la liste des périphériques asservis et les règle pour qu'ils utilisent leur propre signal d'horloge d'échantillonnage.*

A2150-Etalonner (A2150 Calibrate.vi) (Macintosh)

Exécute des échantillonnages d'offset sur les C A/N de l'AT-A2150 spécifié.



Mise en garde *Ce VI n'est pas supporté par les versions NI-DAQ ultérieures à la version 4.9.0 et a été supprimé de la palette d'étalonnage et de configuration. Ce VI est toujours inclus dans la bibliothèque de VIs DAQ pour la compatibilité seulement, c'est la raison pour laquelle ce VI retournera le message : `deviceSupportError`, si vous utilisez la version 5.0 de NI-DAQ ou une version ultérieure. Si vous souhaitez utiliser ce VI, veuillez ré-installer NI-DAQ version 4.9.0 ou une version antérieure.*

Reportez-vous à l'Annexe B, [Capacités matérielles DAQ](#), pour plus d'informations sur le périphérique DAQ AT-A2150.

Lorsque vous lancez LabVIEW, ou lorsque vous réinitialisez l'AT-A2150, LabVIEW effectue un étalonnage de l'offset en utilisant la masse analogique comme référence. N'utilisez ce VI que pour l'étalonnage d'un périphérique avec une référence externe ou pour le ré-étalonnage d'un périphérique concernant la référence de masse après l'utilisation d'une référence externe.

AO-6/10-Etalonner (AO-6/10 Calibrate.vi) (Windows)

Charge un ensemble de constantes d'étalonnage dans les C N/A d'étalonnage ou copie un ensemble de constantes d'étalonnage provenant de l'une des quatre zones de l'EEPROM dans la zone 1 de l'EEPROM.



Vous pouvez charger un ensemble de constantes d'étalonnage existant dans les C N/A d'étalonnage à partir d'une zone de stockage de l'EEPROM intégrée de la carte. Vous pouvez copier les zones de stockage 2 à 5 de l'EEPROM sur la zone de stockage 1. La zone 5 de l'EEPROM contient les constantes d'étalonnage usine. LabVIEW charge automatiquement les constantes d'étalonnage stockées dans la zone 1 de l'EEPROM au moment du démarrage ou lorsque vous réinitialisez le périphérique AT-AO-6/10.



Remarque *Vous pouvez également utiliser l'utilitaire d'étalonnage livré avec le périphérique AT-AO-6/10 pour effectuer une procédure d'étalonnage. Reportez-vous au chapitre sur l'étalonnage dans le Manuel de l'utilisateur du périphérique AT-AO-6/10 pour plus d'informations.*

Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour plus d'informations sur les périphériques DAQ AT-AO-6/10.

Lorsque LabVIEW initialise le périphérique AT-AO-6/10, les constantes d'étalonnage du C A/N stockées dans la **position dans l'EEPROM 1** (zone d'étalonnage de l'utilisateur 1) fournissent les valeurs de gain et d'offset qui assurent un fonctionnement correct du périphérique. Ainsi, cette initialisation revient à exécuter le VI "AO-6/10-Etalonner" (AO-6/10 Calibrate.vi) avec **opération** réglée sur 1 et **position dans l'EEPROM** réglée sur 1. Quand l'AT-AO-6/10 quitte l'usine, **position dans l'EEPROM 1** contient une copie des constantes d'étalonnage stockées dans **position dans l'EEPROM 5** (étalonnage en usine).

Une procédure d'étalonnage exécutée en mode bipolaire n'est pas valable pour le mode unipolaire et réciproquement. Voir le chapitre sur l'étalonnage dans le *Manuel de l'utilisateur du périphérique AT-AO-6/10* pour plus d'informations.

Voies en indice (Channel to Index.vi)

Utilise la configuration de groupe courante pour la tâche spécifiée afin de produire une liste d'indices dans le balayage du groupe ou dans la liste de mise à jour pour chaque voie spécifiée dans la liste des voies.



Vous pouvez utiliser cette liste d'indices de voies pour localiser les données d'une voie particulière dans un buffer de plusieurs voies. Vous pouvez également utiliser les indices pour lire ou écrire dans un sous-ensemble de groupe avec les VIs de lecture et d'écriture.

Reportez-vous aux informations relatives à votre périphérique spécifique à l'Annexe B, *Capacités matérielles DAQ*, pour connaître les limites qui s'appliquent à votre périphérique.

Le Table 10-29-1 présente les valeurs possibles des paramètres **liste des balayages de voies**, **liste des voies** et **indices des voies**. Il présente les valeurs possibles pour les plates-formes Sun. Le paramètre **liste des balayages de voies** est une entrée pour les VIs de configuration de groupe.

Tableau 29-1. Exemples de paramètres pour les VIs “Voies en indice” (Channel to Index.vi)

Liste des balayages de voies	Liste des voies	Indices des voies
1, 3, 4, 5, 7	liste des voies [0] = 5	indices des voies [0] = 3. Les données pour la voie 5 sont en position 3 dans un balayage. Les indices sont de base nulle (premier indice = 0).
1, 3, 4, 5, 7	liste des voies a une longueur de 0.	indices des voies ont une longueur de 0. (Dans ce cas, l'indice est non nul.)
1, 2, 1, 3, 1, 4 (Le périphérique échantillonne la voie 1 trois fois pendant un balayage.)	liste des voies [0] = 1, 1, 1	indices des voies [0] = 0, indices des voies [1] = 2, et indices des voies [2] = 4. La première occurrence de la voie 1 dans un balayage est à l'indice 0, la deuxième à l'indice 2 et la troisième à l'indice 4.
0, 1, 3, 4 (Dans cet exemple, liste des balayages de voies est un groupe d'entrées numériques.)	liste des voies [0] = 3	indices des voies [0] = 2. Les trois bits de données du port sont à l'indice 2 dans la liste de balayage.

Tableau 29-1. Exemples de paramètres pour les VIs “Voies en indice” (Channel to Index.vi) (Suite)

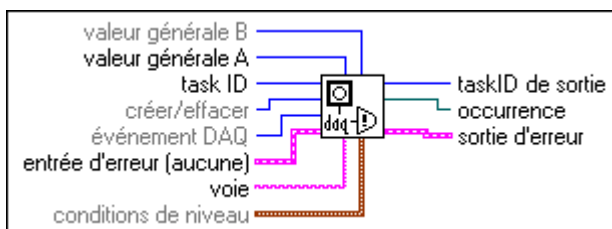
Liste des balayages de voies	Liste des voies	Indices des voies
0:3 (Un AMUX-64T utilisé.)	liste des voies [0] = AM1!9	indices des voies [0] = 9. Les données provenant de la voie 19 sur le périphérique AMUX-64T numéro 1 est à l'indice 9 dans le buffer de données.
SC1!MD1!CH0:7, SC1!MD2!CH0:4	liste des voies [0] = SC1!MD2!CH3	indices des voies [0] = 11. Les données provenant de la voie 3 du module SCXI dans l'emplacement 2 est à l'indice 1 dans le buffer de données.

Tableau 29-2. Exemples de paramètres pour les VIs “Voies en indices” (Channel to Index.vi) sans Sun

Liste de balayage des voies	Liste des voies	Indices des voies
1, 3, 4, 5, 7	liste des voies [0] = 5	indices des voies [0] = 3. Les données de la voie 5 sont à la position 3 dans un balayage. Les indices sont de base zéro.
1, 3, 4, 5, 7	liste des voies a une longueur de 0.	indices de voies ont une longueur de 0. (Dans ce cas, l'état est non nul.)
1, 2, 1, 3, 1, 4 (Le périphérique échantillonne la voie 1 trois fois pendant un balayage.)	liste des voies [0] = 1, 1, 1	indices des voies [0] = 0, indices des voies [1] = 2, et indices des voies [2] = 4. La première occurrence de la voie 1 dans un balayage est à l'indice 0, la deuxième est à l'indice 2 et la troisième à l'indice 4.

Configurer des occurrences DAQ (DAQ Occurrence Config.vi) (Windows)

Crée des occurrences qui sont réglées par des événements d'acquisition de données.



Un événement DAQ peut être la fin d'une acquisition, l'acquisition d'un certain nombre de balayages, un signal analogique conforme à certaines conditions de déclenchement, un événement périodique, un événement (piloté extérieurement) aperiodique, ou une correspondance bonne ou mauvaise du pattern numérique. Votre VI peut être en veille en attendant qu'une occurrence soit définie, permettant à votre ordinateur d'exécuter d'autres VIs.

Quand vous positionnez la commande **créer/réinitialiser** sur 1 (créer) et appelez le VI, ce VI crée une occurrence. Utilisez la commande **événement DAQ** pour sélectionner l'événement qui définit l'occurrence. Câblez l'occurrence que ce VI produit, à la fonction "Attendre une occurrence". Tout ce que vous câblez à la sortie de la fonction "Attendre une occurrence" (Wait on Occurrence) ne s'exécute pas avant que l'occurrence soit définie. L'occurrence est positionnée chaque fois que l'événement se produit. L'occurrence n'est pas réinitialisée tant que vous n'avez pas positionné la commande **créer/réinitialiser** sur 0 (réinitialiser) et appelé ce VI ou appelé le VI "Réinitialiser un périphérique" (Device Reset.vi) pour le périphérique.

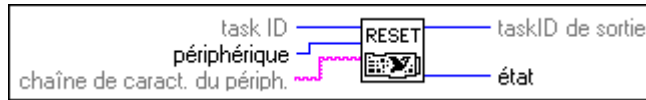
LabVIEW retourne une constante d'E/S sur fichiers "Pas un Refnum" (Not a Refnum) avec un code d'état non nul si l'occurrence ne peut pas être créée.

Pour chaque plate-forme, LabVIEW limite le nombre d'occurrences que vous pouvez définir par seconde. Même si cette limite dépend de la vitesse de votre ordinateur, évitez de dépasser 500 occurrences par seconde.

Pour quelques événements, vous devez exécuter votre opération en utilisant des interruptions au lieu des DMA. Reportez-vous à la description de la commande **événement DAQ** dans cette section pour plus d'informations.

Réinitialiser un périphérique (Device Reset.vi)

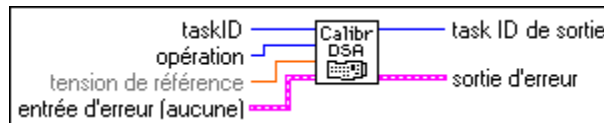
Réinitialise complètement un périphérique ou uniquement la fonction particulière identifiée par **taskID**.



Réinitialiser une fonction **taskID** entraîne les mêmes résultats qu'appeler le VI de contrôle pour cette fonction avec **code de contrôle** positionné sur Réinitialiser. Quand vous réinitialisez complètement le périphérique, LabVIEW réinitialise toutes les tâches et positionne tous les paramètres du périphérique sur leurs valeurs par défaut.

DSA-Etalonner (DSA Calibrate.vi)

Utilisez ce VI pour étalonner votre périphérique DSA.

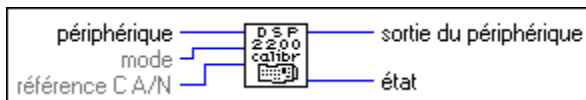


Votre périphérique contient des convertisseurs d'étalonnage N/A (calDAC) qui sont utilisés pour peaufiner l'ensemble des circuits analogiques. Les calDAC doivent être programmés (chargés) avec certains nombres, appelés *constantes d'étalonnage*. Ces constantes sont stockées dans une mémoire rémanente (EEPROM) sur votre périphérique. Pour obtenir une spécification exacte, vous devez exécuter un étalonnage interne de votre périphérique juste avant une série de mesures, mais uniquement après la mise sous tension de votre ordinateur et de votre périphérique et leur avoir laissé au moins 15 minutes pour chauffer. Un étalonnage fréquent produit une performance de mesure plus stable et plus reproductible.

Avant que le périphérique sorte de l'usine, un étalonnage externe est effectué et l'EEPROM contient des constantes d'étalonnage que LabVIEW charge automatiquement dans les calDAC en fonction des besoins. La valeur de la tension de référence de la carte est également stockée dans l'EEPROM et cette valeur est utilisée lorsque vous effectuez ultérieurement un étalonnage automatique. Lorsqu'un étalonnage automatique est effectué, les constantes d'étalonnage sont recalculées et stockées dans l'EEPROM. Lorsque vous effectuez un étalonnage externe, LabVIEW recalcule la valeur de la tension de référence de la carte et effectue ensuite un étalonnage automatique. Cette nouvelle valeur de référence de la carte est utilisée pour toutes les opérations d'étalonnage automatique effectuées par la suite. Si une erreur est faite lors d'un étalonnage externe, vous pouvez à tout moment restaurer l'étalonnage d'usine de la carte pour que la carte puisse être utilisable.

DSP2200-Etalonner (DSP2200 Calibrate.vi) (Windows)

Effectue des étalonnages d'offset sur l'entrée analogique et/ou la sortie analogique du périphérique AT-DSP2200.



Mise en garde *Ce VI n'est pas supporté sur des versions NI-DAQ ultérieures à la version 4.9.0 et a été supprimé de la palette d'étalonnage et de configuration. Ce VI est toujours inclus dans la bibliothèque de VIs DAQ pour des raisons de compatibilité, c'est la raison pour laquelle ce VI retournera le message : `deviceSupportError`, si vous utilisez la version 5.0 de NI-DAQ ou une version ultérieure. Si vous souhaitez utiliser ce VI, veuillez réinstaller NI-DAQ version 4.9.0 ou une version antérieure.*

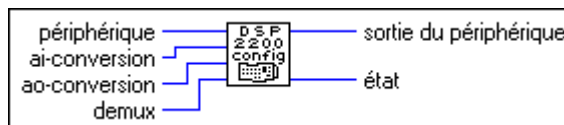
Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour plus d'informations sur le périphérique DAQ AT-DSP2200.

Lorsque vous lancez LabVIEW ou réinitialisez le périphérique AT-DSP2200, LabVIEW effectue un étalonnage de l'offset sur l'entrée et la sortie analogiques en utilisant une masse analogique comme référence.

Vous pouvez utiliser ce VI pour calibrer l'entrée analogique en vous aidant d'une référence externe ou pour ré-étalonner le périphérique AT-DSP2200 pour compenser les modifications de configuration ou d'environnement.

DSP2200-Configurer (DSP2200 Configure.vi) (Windows)

Spécifie la conversion de données et les opérations de démultiplexage que le périphérique AT-DSP2200 réalise sur les données d'entrée et de sortie analogiques.



Mise en garde *Ce VI n'est pas supporté par les versions de NI-DAQ ultérieures à la version 4.9.0 et a été supprimé de la palette d'étalonnage et de configuration. Ce VI est toujours inclus dans la bibliothèque de VIs DAQ pour des raisons de compatibilité uniquement, c'est la raison pour laquelle ce VI retournera le message : `deviceSupportError`, si vous utilisez la version 5.0 de NI-DAQ ou une version ultérieure. Si vous souhaitez utiliser ce VI, veuillez ré-installer NI-DAQ version 4.9.0 ou une version antérieure.*

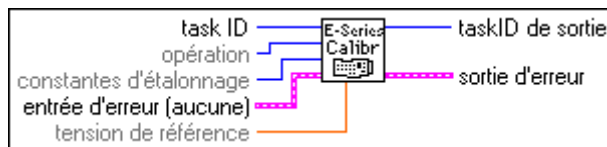
Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour plus d'informations sur le périphérique DAQ AT-DSP2200.

Le logiciel s'exécute localement sur le circuit DSP AT&T WE DSP32C qui lit des données provenant des C A/N et écrit des données dans les C N/A, vous pouvez donc manipuler les données pendant ces conversions. Lorsque vous écrivez des données d'entrées analogiques dans la mémoire DSP, vous pouvez écrire les données sous forme de nombres entiers 16 bits non mis à l'échelle, de nombres à virgule flottante 32C non mis à l'échelle ou de tensions à virgule flottante 32C mises à l'échelle. Vous ne pouvez utiliser l'option **demux** que quand vous écrivez des données d'entrées analogiques dans la mémoire DSP. Lorsque vous activez l'option **demux**, le périphérique écrit des données provenant de la voie 0 consécutivement dans la mémoire DSP, en commençant au début de chaque buffer et écrit les données de la voie 1 consécutivement en commençant au milieu de chaque buffer. Lorsque le périphérique écrit des données d'entrées analogiques dans la mémoire du PC, il peut les écrire sous forme de nombres entiers 16 bits non mis à l'échelle, de nombres à virgule flottante, en simple précision IEEE, non mis à l'échelle ou de tensions en simple précision IEEE mises à l'échelle.

Si le paramètre **ao-conversion** est égal à 0, les données d'origine doivent avoir un format adapté aux C N/A (valeurs du C N/A en nombres entiers 16 bits). Si **ao-conversion** est égal à 1 ou à 3, les données d'origine sont des valeurs du C N/A au format 32C dans la mémoire DSP ou au format simple précision IEEE dans la mémoire du PC. Si **ao-conversion** est égal à 2 ou à 4, les données d'origine sont des tensions au format 32C dans la mémoire DSP ou au format en simple précision IEEE dans la mémoire du PC.

Série E-Etalonner (E-Series Calibrate.vi)

Utilisez ce VI pour étalonner votre périphérique Série E et pour sélectionner un ensemble de constantes d'étalonnage devant être utilisées par LabVIEW.



Mise en garde Lisez le chapitre sur l'étalonnage dans le manuel de l'utilisateur de votre périphérique avant d'utiliser le VI "Série E-Etalonner" (E-Series Calibrate.vi).

Votre périphérique contient des convertisseurs d'étalonnage N/A (calDAC) qui sont utilisés pour peaufiner l'ensemble des circuits analogiques. Les calDAC doivent être programmés (chargés) avec certains nombres, appelés *constantes d'étalonnage*. Ces constantes sont stockées dans une mémoire rémanente (EEPROM) sur votre périphérique ou sont conservées par LabVIEW. Pour obtenir une spécification exacte, vous devez exécuter un étalonnage interne de votre périphérique juste avant une série de mesures, mais uniquement après la mise

sous tension de votre ordinateur et de votre périphérique et leur avoir laissé au moins 15 minutes pour chauffer. Un étalonnage fréquent produit une performance de mesure plus stable et plus reproductible. Vous ne risquez pas d'endommager le périphérique en le ré-étalonnant souvent.

Deux ensembles de constantes d'étalonnage peuvent résider dans deux zones à l'intérieur de l'EEPROM, appelées *zones de chargement*. Un ensemble de constantes est programmé à l'usine, l'autre est laissé à la disposition de l'utilisateur. Une zone de chargement de l'EEPROM correspond à un ensemble de constantes. La zone de chargement que LabVIEW utilise pour charger des calDAC avec des constantes d'étalonnage est appelée la zone de chargement par défaut. Lorsque vous recevez votre périphérique, la zone de chargement par défaut est celle qui contient les constantes d'étalonnage obtenues par l'étalonnage du périphérique en usine. LabVIEW charge automatiquement les constantes d'étalonnage correspondantes stockées dans la zone de chargement la première fois que vous appelez un VI qui les demande.



Remarque *L'étalonnage de votre périphérique Série E est assez long. Ne vous inquiétez pas si l'exécution du VI prend plusieurs secondes.*



Mise en garde *Lorsque vous exécutez ce VI avec le paramètre opération positionné sur auto-étalonnage ou sur étalonnage externe, LabVIEW abandonne toutes les opérations par le périphérique et règle toutes les configurations sur leurs valeurs par défaut. C'est pourquoi, vous devez exécuter ce VI avant tout autre VI DAQ ou lorsqu'aucune autre opération ne s'exécute.*

Périphériques Série E 12 bits

- Connectez la sortie positive de votre source de tension de référence à la voie d'entrée analogique 8.
- Connectez la sortie négative de votre source de tension de référence à la ligne AISENSE.
- Connectez la ligne DAC0 (voie de sorties analogiques 0) à la voie d'entrée analogique 0.
- Si votre source de tension de référence et votre ordinateur sont flottants l'un par rapport à l'autre, connectez la ligne AISENSE à la ligne AIGND ainsi qu'à la sortie négative de votre source de tension de référence.

Périphériques Série E 16 bits

- Connectez la sortie positive de votre source de tension de référence à la voie d'entrée analogique 0.
- Connectez la sortie négative de votre source de tension de référence à la voie de sorties analogiques 8 (en effectuant ces deux connexions, vous fournissez une tension de référence à la voie d'entrée analogique 10 qui est configurée pour une opération différentielle.)
- Si votre source de tension de référence et votre ordinateur sont flottants l'un par rapport à l'autre, connectez la sortie négative de votre source de tension de référence à la ligne ainsi qu'à la voie d'entrée analogique 8.

Obtenir des informations sur un périphérique DAQ (Get DAQ Device Information.vi)

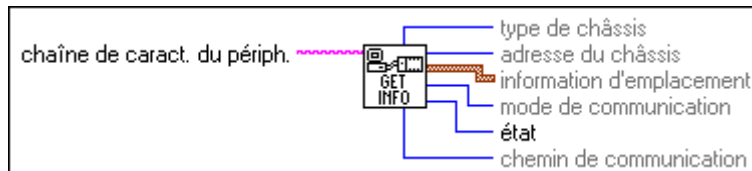
Retourne des informations concernant un périphérique DAQ.



Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour connaître les méthodes de transfert disponibles sur votre périphérique DAQ.

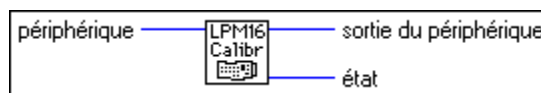
Obtenir des informations sur le matériel SCXI (Get SCXI Information.vi)

Retourne les informations sur la configuration du châssis que vous réglez en utilisant l'utilitaire de configuration ou le VI "Définir les informations d'un matériel SCXI" (Set SCXI Information.vi).



LPM-16-Etalonner (LPM-16 Calibrate.vi)

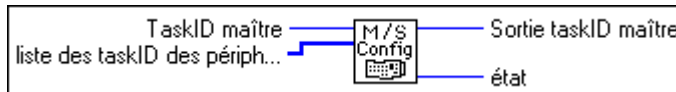
Etalonne le convertisseur PC-LPM-16 ou PC-LPM-16PnP. L'étalonnage calcule la tension d'offset correcte du comparateur de tension, règle la linéarité positive et les erreurs de cadrage intégré entre ± 0.5 LSB chacune, et règle l'erreur du zéro entre ± 1 LSB.



Reportez-vous à l'Annexe B, Capacités matérielles DAQ, pour plus d'informations sur le périphérique PC-LPM-16, DAQCard-500 ou DAQCard-700.

Config. maître/esclave (Master Slave Config.vi)

Configure un périphérique comme un périphérique maître et tous les périphériques restants comme des périphériques esclaves pour des opérations d'entrées analogiques avec plusieurs buffers.



Mise en garde *Ce VI n'est pas supporté par les versions de NI-DAQ ultérieures à la version 4.9.0 et a été supprimé de la palette d'étalonnage et de configuration. Ce VI est toujours inclus dans la bibliothèque de VI DAQ pour des raisons de compatibilité uniquement, c'est la raison pour laquelle ce VI retournera le message : `deviceSupportError`, si vous utilisez la version 5.0 de NI-DAQ ou une version ultérieure. Si vous souhaitez utiliser ce VI, veuillez ré-installer NI-DAQ version 4.9.0 ou une version antérieure.*

Assurez-vous que LabVIEW réactive toujours les périphériques esclaves *avant* le périphérique maître dans une opération d'entrée analogique utilisant plusieurs buffers. Seuls les périphériques suivants, qui supportent des acquisitions avec plusieurs buffers, peuvent utiliser ce VI.

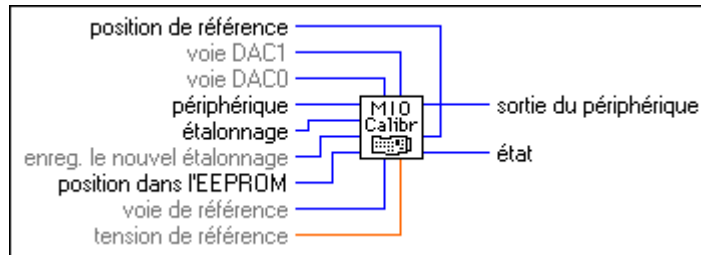
- (Macintosh) NB-A2000, NB-A2100 et NB-A2150.

Le périphérique maître envoie un signal de déclenchement et d'horloge au(x) périphérique(s) esclave(s) pour contrôler l'échantillonnage de ce(s) périphérique(s). Dans le cas d'une acquisition avec plusieurs buffers, vous devez activer le périphérique esclave avant le périphérique maître pour être certain que le périphérique esclave réponde toujours à un signal maître. Si vous activez le périphérique maître en premier, il peut envoyer un signal aux périphériques esclaves avant qu'ils ne puissent répondre. Vous êtes responsable de l'ordre de la configuration initiale. Vous devez toujours démarrer le périphérique maître en dernier. Le VI de configuration maître/esclave s'assure que LabVIEW arme le périphérique maître en dernier pour chaque buffer consécutif pendant une acquisition à buffers multiples.

MIO-Etalonner (MIO Calibrate.vi) (Windows)

Etalonne les valeurs de gain et d'offset des périphériques AT-MIO-16F-5, AT-MIO-64F-5, et AT-MIO-16X pour les C A/N et les C N/A. Vous pouvez exécuter un nouvel étalonnage ou utiliser un ensemble de constantes d'étalonnage existant en copiant les constantes à partir de leur lieu de stockage dans l'EEPROM intégrée. Vous pouvez stocker plusieurs jeux de constantes d'étalonnage. LabVIEW charge automatiquement les constantes d'étalonnage

stockées dans la zone de chargement de l'EEPROM pendant le démarrage ou quand vous réinitialisez le périphérique.



La zone de chargement du périphérique AT-MIO-16F-5 est la zone utilisateur 5. La zone de chargement des périphériques AT-MIO-64F-5 et AT-MIO-16X est la zone utilisateur 8.



Mise en garde *Lisez le chapitre sur l'étalonnage dans le manuel de l'utilisateur de votre périphérique avant d'utiliser le VI "MIO-Etalonner" (MIO Calibrate.vi).*

Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour plus d'informations sur les périphériques DAQ AT-MIO-16F-5, AT-MIO-64F-5 et AT-MIO-16X.



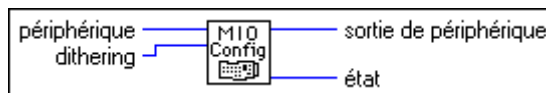
Remarque *Vous devez toujours étalonner le C A/N et les C N/A après avoir étalonné la tension de référence interne.*



Remarque *Si le périphérique prend des mesures d'entrées analogiques et que le mauvais ensemble de constantes d'étalonnage est chargé, vous pouvez obtenir des données erronées.*

MIO-Configurer (MIO Config.vi) (Windows)


Active et désactive le dithering. Ce VI supporte les périphériques suivants : AT-MIO-16F-5, AT-MIO-64F-5, tous les périphériques Série E 12 bits et tous les périphériques de la série 1200.

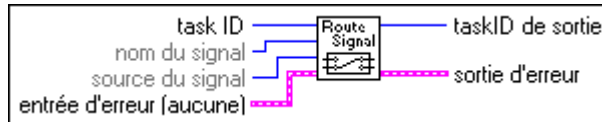


Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour plus d'informations sur les périphériques supportés par ce VI.

Diriger des signaux (Route Signal.vi)

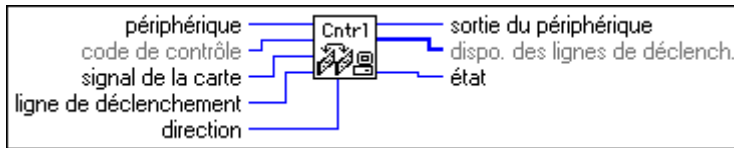
Utilisez ce VI pour diriger un signal interne vers le connecteur d'E/S spécifié ou vers une ligne du bus RTSI, ou pour activer un partage d'horloge sur la ligne d'horloge du bus RTSI.

 **Remarque** Ce VI n'est supporté que par les périphériques des Séries E et 54XX.



Contrôle RTSI (RTSI Control.vi)

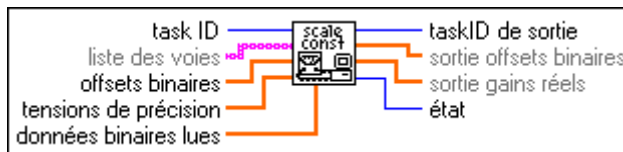
Connecte ou déconnecte les signaux de déclenchement et de séquençement entre les périphériques DAQ le long du bus RTSI (Real-Time System Integration).



Ce VI n'est pas supporté par les périphériques de Série E. Pour les périphériques Série E, des connexions RTSI multiples peuvent être définies directement dans les VIs de compteurs, de sorties analogiques et d'entrée analogiques et être utilisées avec le VI "Diriger des signaux" (Route signal.vi).

Ajuster la constante de mise à l'échelle (Scaling Constant Tuner.vi)

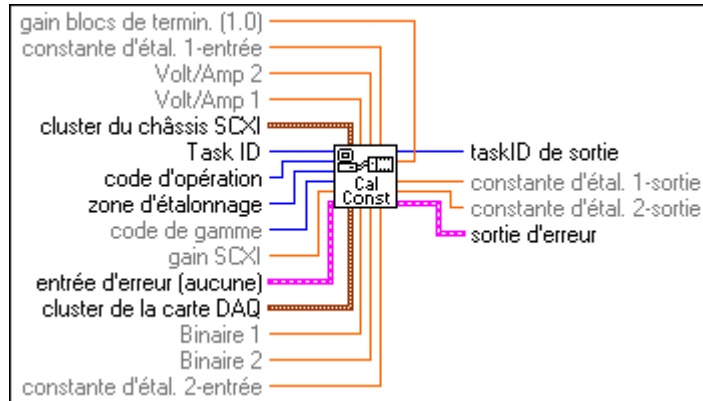
Ajuste les constantes de mise à l'échelle que LabVIEW utilise pour rendre compte de l'offset et du gain non idéal et pour convertir des données binaires d'entrée analogique en tension.



Pour plus d'informations sur ce VI, reportez-vous à sa description au chapitre 30, [Vls de conditionnement de signaux](#).

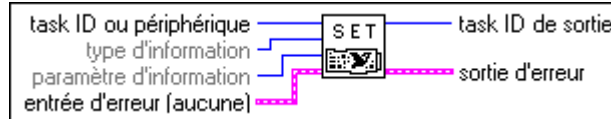
Constantes d'étalonnage SCXI (SCXI Cal Constants.vi)

Calcule les constantes d'étalonnage de la voie et de la gamme données ou du gain à l'aide de paires tension/binaire mesurées. Vous pouvez utiliser ce VI avec n'importe quel module SCXI.



Définir les infos d'un périphérique DAQ (Set DAQ Device Information.vi)

Définit le mode de transfert de données pour différents types d'opérations.



Reportez-vous à l'Annexe B, *Capacités matérielles DAQ*, pour connaître les méthodes de transfert disponibles sur votre périphérique DAQ.

Définir les infos d'un matériel SCXI (Set SCXI Information.vi)

Définit les informations sur la configuration de châssis SCXI.



Utilisez ce VI pour remplacer la configuration déjà définie avec l'utilitaire de configuration. Vous pouvez utiliser ce VI *au lieu de* l'utilitaire de configuration pour entrer les informations de configuration de châssis. Si vous n'utilisez pas ce VI, le premier VI qui accède à un châssis SCXI tente automatiquement de charger des informations du fichier de configuration.

VIs de configuration de voies

L'illustration suivante présente la palette des VIs de configuration de voies.



Obtenir les noms de voies DAQ (Get DAQ Channel Names.vi)

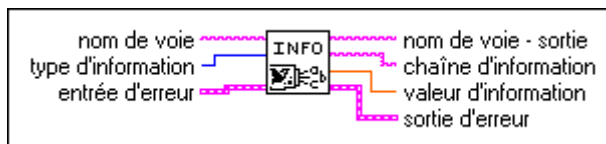
Retourne un tableau de tous les noms de voies du fichier de configuration par défaut. Un tableau correspondant des unités physiques configurées pour chaque voie est également retourné. A l'aide de **type de voie**, vous pouvez choisir de récupérer toutes les voies, ou uniquement les voies d'entrée et de sorties analogiques ou d'E/S numériques.



Remarque *Ce VI est particulier aux ordinateurs utilisant NI-DAQ 5.0 ou une version plus récente. LabVIEW retourne un message d'erreur indiquant que le VI n'est pas supporté si vous tentez d'utiliser ce VI sur des ordinateurs qui ne possèdent pas NI-DAQ 5.0 ou une version plus récente.*

Obtenir des infos de voies (Get Channel Information.vi)

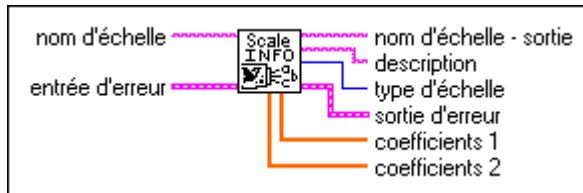
Retourne des informations de configuration sur une voie configurée dans l'Assistant d'Entrées/Sorties DAQ.



Remarque *Ce VI est particulier aux ordinateurs utilisant NI-DAQ 5.0 ou une version plus récente. LabVIEW retourne un message d'erreur indiquant que le VI n'est pas supporté si vous tentez d'utiliser ce VI sur des ordinateurs qui ne possèdent pas NI-DAQ 5.0 ou une version plus récente.*

Obtenir des infos d'échelle (Get Scale Information.vi)

Retourne des informations de configuration d'une échelle configurée dans l'Assistant Entrées/Sorties DAQ.



Remarque

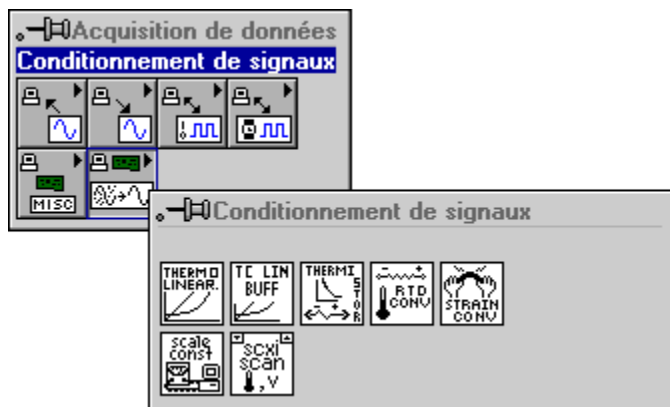
Ce VI est particulier aux ordinateurs utilisant NI-DAQ 5.0 ou une version plus récente. LabVIEW retourne un message d'erreur indiquant que le VI n'est pas supporté si vous tentez d'utiliser ce VI sur des ordinateurs qui ne possèdent pas NI-DAQ 5.0 ou une version plus récente.

VIs de conditionnement de signaux

Ce chapitre décrit les VIs de conditionnement de signaux d'acquisition de données que vous utilisez pour convertir des tensions d'entrées analogiques lues à partir de détecteurs de température à résistance (RTD), de jauges de contrainte ou de thermocouples en unités de contrainte ou de température.

Vous pouvez modifier les formules de conversion utilisées dans ces VIs ou les remplacer par les vôtres pour répondre aux exigences spécifiques de votre application. Si vous modifiez ou remplacez les formules, vous devez sauvegarder le nouveau VI dans un de vos propres répertoires ou dossiers différents de `vi.lib`.

Vous pouvez accéder aux VIs de conditionnement de signaux en choisissant **Fonctions**»**Acquisition de données**»**Conditionnement de signaux**, comme illustré ci-dessous.

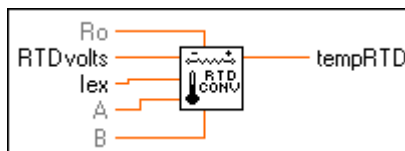


Description des VIs de conditionnement de signaux

Les VIs de conditionnement de signaux disponibles sont les suivants.

Convertir une mesure de RTD (Convert RTD Reading.vi)

Convertit une tension que vous lisez grâce à un RTD en une température en Celsius.



Ce VI calcule d'abord la résistance du RTD en divisant la **tension RTD** par **Iex**. Le VI convertit ensuite la résistance en température à l'aide de l'équation de Callendar Van-Dusen pour les RTD suivants :

$$R_t = R_0[1 + At + Bt^2 + C(t-100)t^3]$$

Pour les températures supérieures à 0° C, le coefficient C est égal à 0, et l'équation précédente se réduit à une équation quadratique pour laquelle l'algorithme implémenté dans le VI donne la racine appropriée. Ainsi, ce VI de conversion n'est exact que pour des températures supérieures à 0° C.

La documentation de votre RTD doit vous donner **R₀** et les coefficients **A** et **B** pour l'équation de Callendar Van-Dusen. Les RTD les plus courants sont les RTD au platine 100 Ω qui suivent la courbe de température européenne (DIN 43760) ou américaine. Le tableau suivant donne les valeurs **A** et **B** pour les courbes européennes et américaines.

Courbe européenne (DIN 43760)	Courbe américaine
A = 3.90802e-03	A = 3.9784e-03
B = -5.80195e-07	B = -5.8408e-07
(α = 0.00385; δ = 1.492)	(α = 0.00392; δ = 1.492)

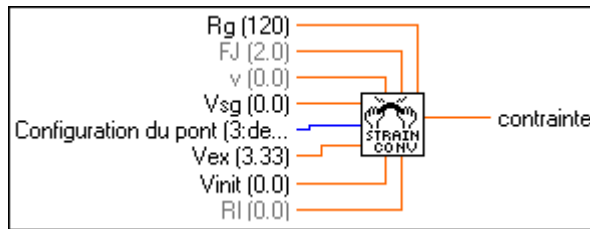
Quelques documentations de capteurs RTD donnent les valeurs de α et δ, à partir desquelles vous pouvez calculer **A** et **B** avec les équations suivantes :

$$A = \alpha(1 + \delta/100)$$

$$B = -\alpha\delta/100^2$$

Convertir une mesure de jauge de contrainte (Convert Strain Gauge Reading.vi)

Convertit une tension que vous lisez grâce à une jauge de contrainte en unités de contrainte.



La formule de conversion que le VI utilise est uniquement basée sur la configuration du pont. Les Figures 30-1 à 30-3 illustrent les sept configurations de pont que vous pouvez utiliser et les formules correspondantes. Pour toutes les configurations de pont, le VI utilise la formule suivante pour obtenir V_r :

$$V_r = (V_{sg} - V_{init}) / V_{ex}$$

Dans les diagrammes de circuit, V_{OUT} est la tension que vous mesurez et passez au VI de conversion en tant que paramètre V_{jc} . Dans les configurations quart de pont et demi-pont, R_1 et R_2 sont des résistances fictives qui ne sont pas directement incorporées dans la formule de conversion. Les modules SCXI-1121 et SCXI-1122 fournissent R_1 et R_2 pour un réseau de commandes par pont, si nécessaire.

Reportez-vous au *Getting started with SCXI* pour plus d'informations sur les réseaux de commande par pont et l'excitation de tension.

Les Figures 30-1 à 30-3 illustrent les réseaux de commande par pont disponibles.

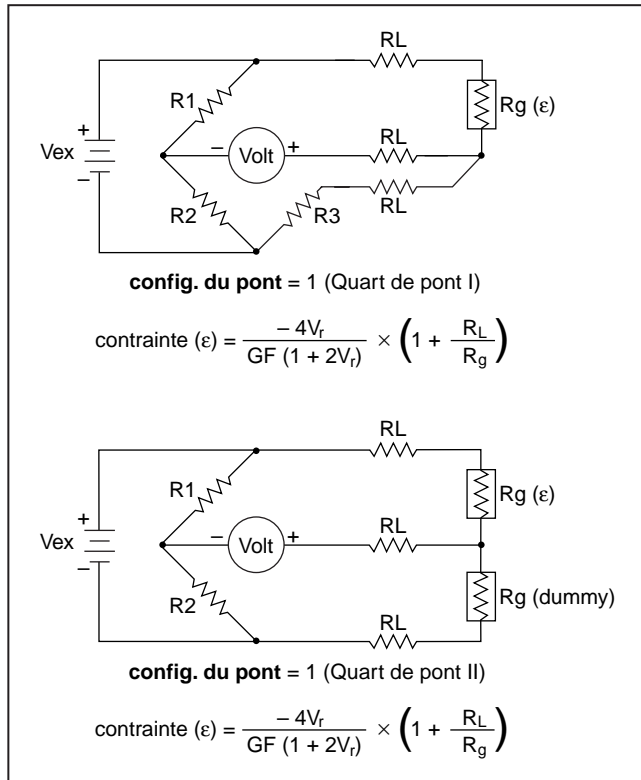


Figure 30-1. Réseaux de commande par pont pour jauge de contrainte (Configuration quart de pont)

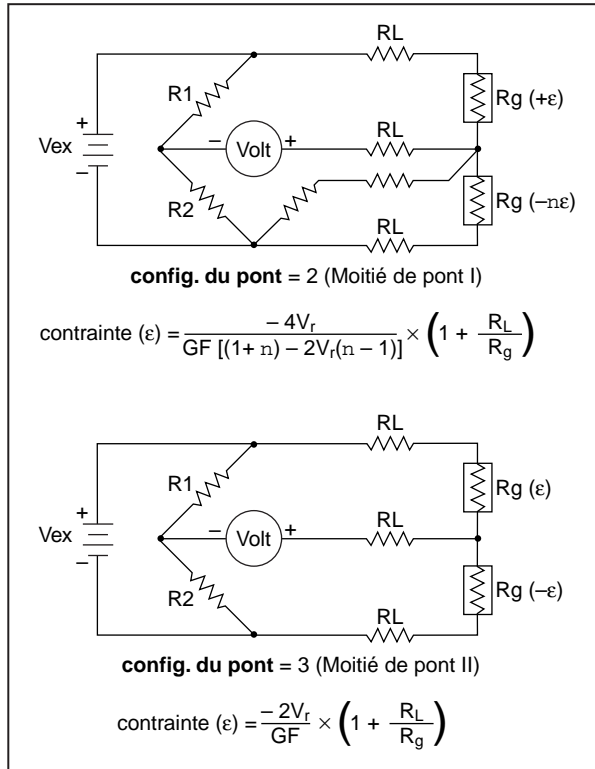


Figure 30-2. Réseaux de commande par pont pour jauge de contrainte (Configuration demi-pont)

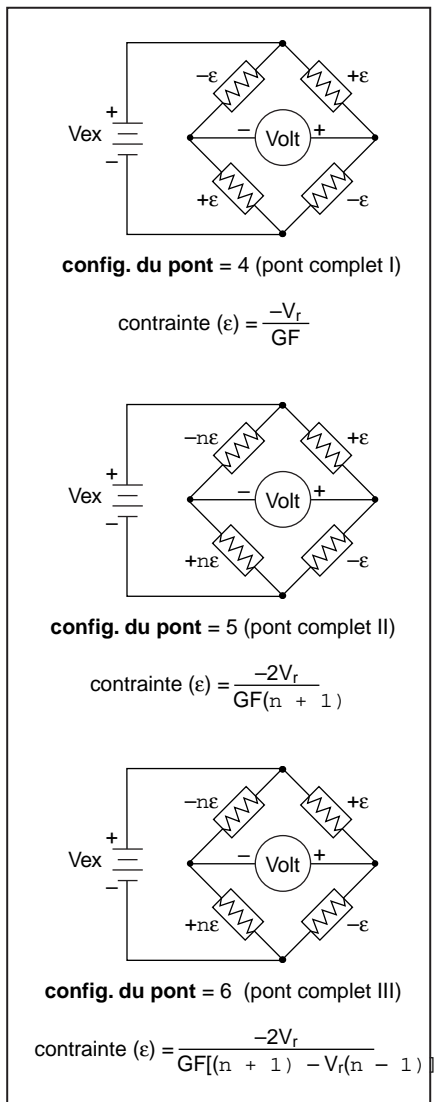
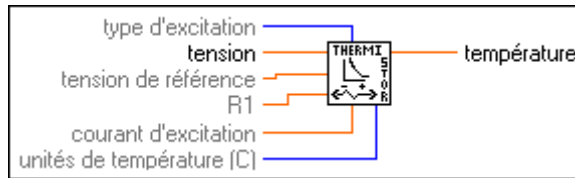


Figure 30-3. Réseaux de commande par pont pour jauge de contrainte (Configuration pont intégral)

Convertir une mesure de thermistance (Convert Thermistor Reading.vi)

Convertit la tension d'une thermistance en température. Ce VI a deux modes d'opération différents pour des thermistances excitées par tension ou par courant.



Ce VI a deux modes d'opération à utiliser suivant les types de circuits à thermistance. La Figure 30-4 illustre une manière de connecter la thermistance à une référence de tension. Il s'agit de la configuration utilisée dans les blocs de terminaux SCXI-1303, SCXI-1322, SCXI-1327 et SCXI-1328, qui utilise une thermistance intégrée pour la compensation de soudure froide.

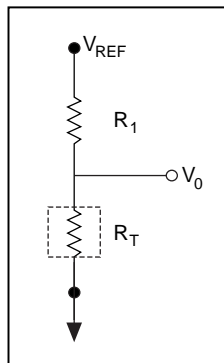


Figure 30-4. Diagramme du circuit d'une thermistance dans un diviseur de tension

La Figure 30-5 illustre un circuit dans lequel la thermistance est excitée par une source de courant constante. Un exemple de cette configuration est l'utilisation du périphérique DAQPad-MIO-16XE-50, qui fournit une sortie de courant constant. Le périphérique DAQPad-TB-52 comporte une thermistance pour détecter une soudure froide.

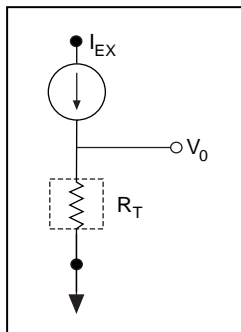


Figure 30-5. Diagramme du circuit d'une thermistance avec excitation de courant

Si la thermistance est excitée par tension, l'équation suivante illustre la relation entre la résistance de la thermistance, R_T et les valeurs d'entrée :

$$R_T = R_I \left(\frac{V_0}{V_{REF} - V_0} \right)$$

Si la thermistance est excitée par courant, l'équation est :

$$R_T = \frac{V_0}{I_{EX}}$$

L'équation suivante est la formule standard que le VI utilise pour convertir une résistance de thermistance en température :

$$T_K = \frac{I}{a + b(\ln R_T) + c(\ln R_T)^3}$$

Les valeurs a , b et c utilisées par ce VI sont données ci-dessous. Ces valeurs sont correctes pour les thermistances fournies sur les blocs de terminaux de SCXI et DAQPad-TB-52. Si vous utilisez une thermistance avec des valeurs a , b et c différentes (reportez-vous aux caractéristiques de votre thermistance), modifiez le diagramme du VI pour utiliser vos propres valeurs a , b et c .

$$a = 1.295361E-3$$

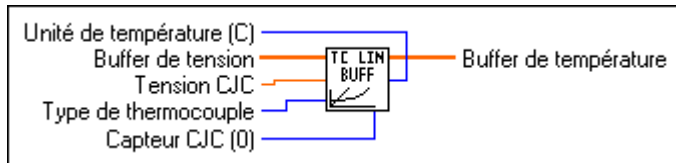
$$b = 2.343159E-4$$

$$c = 1.018703E-7$$

Le VI donne une température exprimée en degrés Celsius. En conséquence, $T_C = T_K - 273.15$.

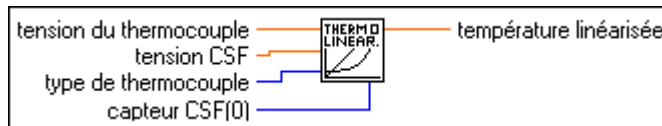
Convertir un buffer de thermocouple (Convert Thermocouple Buffer.vi)

Convertit un buffer de tension mesuré par un thermocouple en un buffer de température exprimé en degrés Celsius.



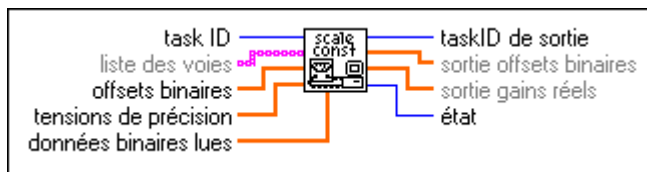
Convertir une mesure de thermocouple (Convert Thermocouple Reading.vi)

Convertit une tension mesurée par un thermocouple en une valeur de température exprimée en degrés Celsius.



Ajuster la constante de mise à l'échelle (Scaling Constant Tuner.vi)

Ajuste les constantes de mise à l'échelle que LabVIEW utilise pour calculer l'offset et le gain non idéal et pour convertir les données binaires d'entrées analogiques en données de tension.



Pour utiliser correctement ce VI, vous devez d'abord prendre deux mesures d'entrée analogiques : une mesure d'offset pour une entrée nulle et une mesure de tension connue.

L'offset binaire par défaut pour chaque voie du groupe est 0. Pour déterminer l'offset binaire réel pour un chemin de voie, mettez les entrées de voie à la masse et prenez une mesure binaire ou prenez plusieurs données binaires et faites-en la moyenne pour obtenir des LSB de l'offset.

Si vous utilisez un module SCXI, mettez les entrées des voies du SCXI à la masse pour mesurer l'offset du chemin complet du signal, comprenant à la fois le module SCXI et le périphérique DAQ. Les modules SCXI-1100, SCXI-1122 et SCXI-1141 comportent un commutateur interne que vous pouvez utiliser pour mettre à la masse les entrées d'amplificateur sans avoir à câbler les terminaux à la masse. Pour utiliser cette fonction, entrez la chaîne de caractères `CALGND` particulière au SCXI dans la chaîne de voie de votre SCXI décrite dans la section *Offset de l'amplificateur* du chapitre 21, *Applications SCXI communes*, dans le *Manuel de base d'acquisition de données LabVIEW*. Utilisez des VIs d'entrées analogiques intermédiaires ou avancés pour obtenir des données binaires plutôt que des données de tension.



Remarque *Si votre périphérique supporte le dithering, activez le dithering de votre périphérique DAQ lorsque vous prenez plusieurs mesures et que vous en faites une moyenne.*

LabVIEW suppose que les paramètres de gain des périphériques DAQ et des modules SCXI sont idéaux quand il met les données binaires à l'échelle de tension, sauf si vous utilisez ce VI pour déterminer les valeurs de gain réel des voies. Appliquez une tension de précision connue à chacune des voies et prenez une mesure binaire ou, effectuez plusieurs mesures pour chaque voie et calculez une mesure binaire moyenne pour chacune d'elles. Votre tension de précision doit être dix fois plus précise que la résolution de votre périphérique DAQ pour obtenir des résultats significatifs. Lorsque vous connectez les paramètres **données binaires lues**, **tensions de précision** et **offsets binaires** à ce VI, LabVIEW détermine le gain réel à l'aide de la formule suivante :

$$\text{gain réel} = \frac{(\text{résolution de tension}) * (\text{données binaires lues} - \text{offset binaire})}{\text{tension de précision}}$$

Dans cette formule, la valeur **résolution de tension** est exprimée en volts par LSB et varie en fonction du type de périphérique DAQ, les réglages de polarité et de la gamme d'entrée. Par exemple, la résolution de tension d'un périphérique PCI-MIO-16E-1 en mode bipolaire avec une gamme d'entrée de +5 à -5 V est 2,44 mV. Le VI retourne un tableau des valeurs de gain réel que le VI stocke pour chaque voie.



Remarque *Quand vous prenez des mesures pour déterminer l'offset et le gain réel, vous devez utiliser les mêmes paramètres de limites d'entrée et fréquences d'horloge que ceux dont vous vous servez pour mesurer vos signaux d'entrée.*

LabVIEW utilise l'équation suivante pour transformer des données binaires en tension :

$$\text{tension} = \frac{\text{résolution de tension} * (\text{données binaires lues} - \text{offset binaire})}{\text{gain}}$$

Lorsque vous exécutez le VI “AI-Configurer un groupe” (AI Group Config.vi), il positionne les attributs de toutes les voies du groupe sur leurs valeurs par défaut, y compris les valeurs d’offset binaire et de gain.

Câblez la **liste de voies** si vous souhaitez ajuster les constantes de mise à l’échelle d’un sous-ensemble de voies du groupe. Si vous ne câblez pas la **liste des voies**, le VI ajuste les constantes de mise à l’échelle de toutes les voies du groupe. Le VI utilise la même méthode que le VI “AI-Configurer le matériel” (AI Hardware Config.vi) pour appliquer des valeurs dans les tableaux d’entrée d’**offset binaires**, de **tensions de précision** et de **données binaires**. Ainsi, si vous avez câblé la **liste des voies** sur ce VI, les premières valeurs (à l’indice 0) des tableaux d’entrée (**offset binaires**, de **tensions de précision** et de **données binaires**) s’appliquent aux voies listées à l’indice 0 de la **liste des voies**. Si vous ne câblez pas la **liste des voies**, les premières valeurs des tableaux d’entrée s’appliquent à la première voie du groupe. Le VI applique les valeurs de chaque tableau d’entrée aux voies de la **liste des voies** ou au groupe de cette façon, jusqu’à ce que le VI remplisse les tableaux. Si des voies de la **liste des voies** ou du groupe ne sont pas encore configurées, le VI leur applique des valeurs finales des tableaux.

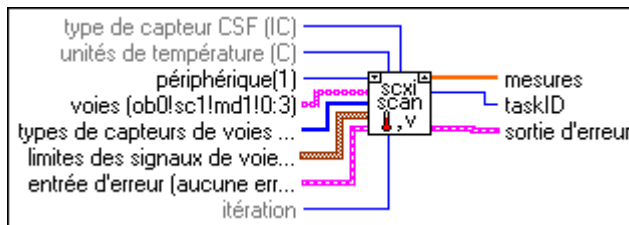
Si vous souhaitez ajuster seulement les offsets des voies et assumer que les paramètres de gain du périphérique DAQ et les modules SCXI sont idéaux, câblez uniquement le paramètre **offsets binaires** et ne câblez pas **tensions de précision** ni **données binaires**.

Vous pouvez également utiliser ce VI pour récupérer les valeurs d’offset binaire et de gain réel de toutes les voies du groupe en ne câblant que **taskID**.

Une fois que vous avez utilisé ce VI pour ajuster les constantes de mise à l’échelle d’un chemin de voie, tout VI d’entrée analogique qui retourne des données de tension, utilise ces constantes lors d’une mise à l’échelle. Utilisez le VI “AI-Configurer un groupe” (AI Group Config.vi) pour réinitialiser les constantes de mise à l’échelle de chaque voie du groupe à leurs valeurs par défaut (offset à zéro et gain idéal).

SCXI-Balayage de température (SCXI Temperature Scan.vi)

Ce VI retourne un balayage de données de température unique provenant d’une liste de voies SCXI. Ce VI utilise un moyennage pour réduire les bruits à 60 Hz et 50 Hz et effectue une linéarisation de thermocouple et une compensation d’offset du module SCXI-1100.



Partie III

Fonctions et VIs d'E/S d'instruments

La Partie III, *Fonctions et VIs d'E/S d'instruments*, décrit les drivers d'instrument de LabVIEW ainsi que les VIs et fonctions GPIB, port série, modèles de driver d'instrument et VISA. Cette partie contient les chapitres suivants :

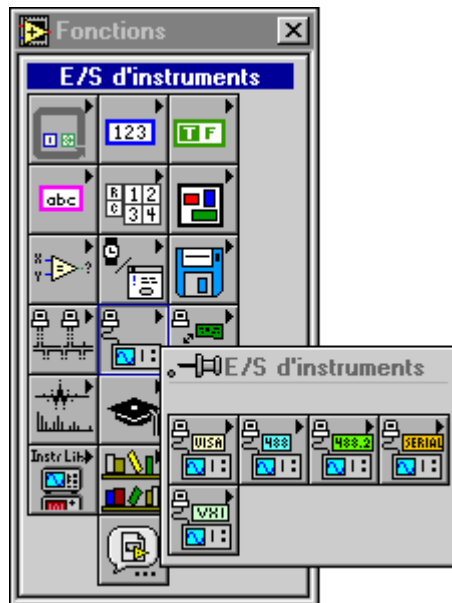
- Le chapitre 31, *Introduction aux VIs d'E/S d'instruments LabVIEW*, décrit les drivers d'instrument de LabVIEW ainsi que les VIs et fonctions GPIB, port série, modèles de driver d'instrument et VISA.
- Le chapitre 32, *Modèles de VIs de drivers d'instruments*, décrit les VIs Modèles de driver d'instrument.
- Le chapitre 33, *Référence de la bibliothèque VISA*, décrit les opérations et les attributs de la bibliothèque de référence VISA.
- Le chapitre 34, *Fonctions GPIB traditionnelles*, décrit les fonctions traditionnelles.
- Le chapitre 35, *Fonctions GPIB 488.2*, décrit les fonctions IEEE 488.2 (GPIB).
- Le chapitre 36, *VIs de Port série*, décrit les VIs utilisés pour les opérations sur port série.

Introduction aux VIs d'E/S d'instruments LabVIEW

Ce chapitre décrit les drivers d'instruments de LabVIEW, le modèle de driver d'instrument GPIB, port série, les VIs VISA et leurs fonctions.

Les VIs de drivers d'instruments sont dans la palette **Fonctions** du diagramme dans LabVIEW. Les VIs de drivers d'instruments sont situés en bas de la palette **Fonctions**.

Pour accéder à la palette **E/S d'instruments**, choisissez **Fonctions**» **E/S d'instruments**, comme illustré ci-dessous.



La palette **E/S d'instruments** est composée des sous-palettes suivantes :

- VISA
- GPIB traditionnel
- GPIB 488.2
- Série

Grâce à la fenêtre Aide de LabVIEW (**Aide**»**Visualiser l'aide**), vous trouverez des informations utiles sur des VIs particuliers. Quand vous placez le curseur sur une icône de VI, le diagramme de câblage et les noms des paramètres de ce VI apparaissent dans la fenêtre d'aide. Vous pouvez également trouver des informations sur les commandes et indicateurs de la face-avant en plaçant votre curseur sur la commande ou l'indicateur avec la fenêtre d'aide ouverte. Pour plus d'informations sur la fenêtre Aide de LabVIEW, reportez-vous à la section *Obtention d'aide* du chapitre 1, *Introduction à la programmation en G*, du *Manuel de référence de la programmation en G*.

En plus de la fenêtre Aide, LabVIEW vous offre des informations en ligne plus étendues. Pour accéder à ces informations, sélectionnez **Aide**»**Référence en ligne**. Pour accéder à la description en ligne de la plupart des objets du diagramme, sélectionnez **Référence en ligne** dans le menu déroulant des objets. Pour plus d'informations sur la création de vos propres fichiers de référence en ligne, voir la section *Création de vos propres fichiers d'aide* au chapitre 5, *Impression et documentation des VIs*, du *Manuel de référence de la programmation en G*.

Vue d'ensemble des drivers d'instruments

Un driver d'instrument LabVIEW est un ensemble de VIs qui contrôlent un instrument programmable. Chaque VI correspond à une opération contrôlée par programme telle que configurer, lire, écrire ou déclencher l'instrument. Les drivers d'instruments LabVIEW simplifient le contrôle d'instruments et réduisent la durée de développement de programmes de test en éliminant la nécessité d'apprendre le protocole de programmation de bas niveau pour chaque instrument.

La bibliothèque de drivers d'instruments LabVIEW de National Instruments contient des drivers d'instruments destinés à plusieurs instruments programmables, dont les instruments GPIB, VXI et Série. Si un driver approprié à votre instrument est dans la bibliothèque, utilisez-le tel quel pour contrôler votre instrument. Les drivers d'instruments sont distribués avec un code source du diagramme pour que vous puissiez les

adapter à votre application, si nécessaire. S'il n'existe pas de driver pour votre instrument, essayez l'une des suggestions suivantes :

- Utilisez un driver pour un instrument similaire. Souvent, les instruments semblables d'un même fabricant ont des drivers d'instruments similaires, voire identiques.
- Modifiez les VIs de modèle de driver d'instrument pour créer un nouveau driver pour votre instrument.
- Utilisez les bibliothèques d'E/S d'instruments GPIB, VXI, Série ou VISA livrées avec LabVIEW pour envoyer et recevoir directement des commandes à et de votre instrument.
- Reportez-vous au chapitre 7, *Utiliser un driver d'instrument de LabVIEW*, dans le *Manuel de l'utilisateur LabVIEW* pour des informations sur l'utilisation des drivers d'instruments LabVIEW de National Instruments.

Distribution d'un driver d'instrument

Les drivers d'instruments LabVIEW sont distribués de plusieurs façons, y compris électroniquement, via un service télématique, Internet et sur CD-ROM.

Vous pouvez télécharger les versions les plus récentes des drivers d'instruments LabVIEW à partir de l'un des services télématiques de National Instruments et, si vous avez accès à Internet, vous pouvez télécharger les fichiers les plus récents de drivers d'instruments du site FTP de National Instruments. Voir les sections Support du service télématique et Support FTP de l'Annexe D, *Informations à l'attention du client*.

Distribution d'un driver d'instrument sur CD-ROM

La bibliothèque complète des drivers d'instruments LabVIEW est disponible sur CD-ROM. Les drivers d'instruments sur CD-ROM sont disponibles gratuitement auprès de National Instruments.

Vous pouvez obtenir la liste des drivers d'instruments la plus récente en utilisant un téléphone à touches en composant le (001-512) 418-1111 aux Etats-Unis, qui est le numéro du système de télécopie automatisé de National Instruments, Fax-on-demand (télécopie sur demande) ou en appelant National Instruments.

VIs de modèle de drivers d'instruments

Les modèles de driver d'instrument LabVIEW constituent la base du développement de tous les drivers d'instruments LabVIEW. Les modèles ont une structure simple et souple et un ensemble de VIs de drivers d'instruments commun que vous pouvez utiliser pour le développement de drivers. Les VIs établissent un format standard pour tous les drivers LabVIEW et chacun de ces VIs reçoit des instructions pour le modifier et l'adapter à un instrument particulier.

Les modèles de drivers d'instruments LabVIEW sont des VIs de drivers d'instruments prédéfinis qui exécutent des opérations communes telles que l'initialisation, le test automatique, la remise à zéro, la consultation d'erreur et ainsi de suite. Plutôt que de développer vos propres VIs pour accomplir ces tâches, utilisez les modèles de drivers d'instruments qui sont déjà conformes aux standards LabVIEW.

Le chapitre 32, *Modèles de VIs de drivers d'instruments*, fournit plus d'informations sur les modèles de driver d'instrument.

Introduction à la bibliothèque VISA

VISA (Virtual Instrument Software Architecture [Architecture logicielle d'instrument virtuel]) est une bibliothèque d'interfaces simples destinée à contrôler des instruments VXI, GPIB, RS-232 et autres types d'instruments. La bibliothèque VISA fournit un ensemble standard de programmes d'E/S utilisés par tous les drivers d'instruments LabVIEW. En vous aidant des fonctions VISA, vous pouvez construire un VI de driver d'instrument simple qui contrôle un modèle d'instrument particulier à travers des interfaces d'E/S différentes. Le transfert d'une chaîne descripteur d'instrument à la fonction VISA Open permet de sélectionner quel type d'E/S est utilisé pour communiquer avec l'instrument. Une fois que la session avec l'instrument est ouverte, les fonctions telles que VISA Read et VISA Write exécutent les activités d'E/S génériques de telle sorte que le programme ne soit pas lié à des fonctions GPIB ou VXI spécifiques. Un tel driver d'instrument est indépendant de toute interface et peut être utilisé tel quel dans différents systèmes.

Les drivers d'instruments qui utilisent les fonctions VISA exécutent des activités particulières à l'instrument et non pas à l'interface de communication. Ceci crée plus d'opportunités pour utiliser le driver d'instrument dans diverses situations.

Pour plus d'informations sur les fonctions VISA, voir le chapitre 33, [Référence de la bibliothèque VISA](#).

Introduction au GPIB

Le bus GPIB (General Purpose Interface Bus) est un lien ou un système d'interface par lequel des périphériques électroniques interconnectés communiquent.

Fonctions GPIB traditionnelles de LabVIEW

Ces fonctions GPIB sont compatibles avec les périphériques IEEE 488 et IEEE 488.2 et suffisent à la plupart des applications. Pour des applications complexes plus nombreuses telles que l'utilisation de plusieurs périphériques et de plusieurs interfaces GPIB, utilisez les fonctions GPIB IEEE 488.2.

Pour plus d'informations sur les fonctions GPIB traditionnel de LabVIEW, voir le chapitre 34, [Fonctions GPIB traditionnelles](#).

Fonctions GPIB 488.2

L'utilisation des fonctions GPIB 488.2 avec des périphériques compatibles IEEE 488.2 améliore la prévisibilité du comportement de l'instrument et du logiciel et réduit les différences de programmation entre les instruments de différents fabricants.

Les révisions les plus récentes de plusieurs cartes GPIB de National Instruments sont entièrement conformes aux normes IEEE 488.2 pour les contrôleurs. Le package LabVIEW comprend également des fonctions qui utilisent la norme IEEE 488.2. En utilisant ces fonctions, votre interface de programmation adhère strictement au standard IEEE 488.2 pour des séquences de commandes et de données.

Les fonctions GPIB 488.2 ont la même fonctionnalité que celle du GPIB traditionnel et comprennent les améliorations et les ajouts suivants :

- Spécifiez l'adresse du périphérique GPIB par un nombre entier plutôt que par une chaîne de caractères. Spécifiez ensuite le numéro du bus par une commande numérique supplémentaire qui facilite la gestion d'interfaces GPIB multiples.
- Déterminez l'état, l'erreur et/ou le nombre d'octets GPIB, directement à partir du cadre connecteur de chaque fonction GPIB 488.2.

L'utilisation de la fonction GPIB Status n'est plus nécessaire pour obtenir des informations d'erreur ou autres.

- La fonction FindLstn implémente le protocole Find All Listeners de IEEE 488.2. Utilisez cette fonction au début d'une application pour déterminer quels périphériques sont présents sur le bus sans connaître leurs adresses.
- La fonction GPIB Misc est toujours disponible mais elle n'est plus nécessaire dans la plupart des cas. La norme IEEE 488.2 spécifie les programmes pour la plupart des besoins d'application du GPIB qui sont implémentés comme des fonctions. Toutefois, vous pouvez associer la fonction GPIB Misc ainsi que d'autres fonctions GPIB avec les fonctions du GPIB 488.2, si cela est nécessaire.
- Certaines fonctions du GPIB 488.2 ont une fonctionnalité de bas niveau et d'autres de haut niveau, pour une adaptation à n'importe quelle application GPIB. Utilisez les fonctions de bas niveau dans des situations sans contrôleur ou lorsque vous avez besoin de plus de souplesse.
- Même si vous devez utiliser un contrôleur compatible IEEE 488.2 avec ces fonctions, celles-ci peuvent contrôler à la fois les périphériques IEEE 488.1 et IEEE 488.2. Les fonctions GPIB 488.2 sont divisées en cinq catégories : périphérique unique, plusieurs périphériques, gestion de bus, bas niveau et général.

Fonctions avec périphérique unique

Les fonctions avec périphérique unique exécutent des opérations d'E/S GPIB et de contrôle avec un périphérique GPIB unique. En général, chaque fonction accepte une adresse de périphérique unique comme l'une de ses entrées.

Pour plus d'informations sur les fonctions avec périphérique unique, reportez-vous au chapitre 35, *Fonctions GPIB 488.2*.

Fonctions avec plusieurs périphériques

Les fonctions avec plusieurs périphériques exécutent des opérations d'E/S GPIB et de contrôle avec plusieurs périphériques GPIB à la fois. En général, chaque fonction accepte un tableau d'adresse comme l'une de ses entrées.

Pour plus d'informations sur les fonctions avec plusieurs périphériques reportez-vous au chapitre 35, *Fonctions GPIB 488.2*.

Fonctions de gestion de bus

Les fonctions de gestion de bus exécutent des fonctions système ou rapportent l'état du système.

Pour plus d'informations sur les fonctions de gestion de bus, reportez-vous au chapitre 35, [Fonctions GPIB 488.2](#).

Fonctions de bas niveau

Grâce aux fonctions de bas niveau, vous pouvez créer un programme plus spécifique et détaillé qu'avec les fonctions de haut niveau. Utilisez des fonctions de bas niveau dans des situations inhabituelles ou qui requièrent plus de souplesse.

Pour plus d'informations sur les fonctions de bas niveau, reportez-vous au chapitre 35, [Fonctions GPIB 488.2](#).

Fonctions générales

Pour plus d'informations sur les fonctions générales, reportez-vous au chapitre 35, [Fonctions GPIB 488.2](#).

Vue d'ensemble des VIs de port série

Les VIs de port série configurent le port série de votre ordinateur et pilotent les E/S qui utilisent ce port.

Pour plus d'informations sur les fonctions de port série, reportez-vous au chapitre 36, [VIs de Port série](#).

Modèles de VIs de drivers d'instruments

Ce chapitre décrit les modèles de VIs de drivers d'instruments. Ces VIs se trouvent dans `examples\instr\insttmpl.llb`.

Introduction aux modèles de VIs de drivers d'instruments

Les modèles de drivers d'instruments LabVIEW constituent la base de toute l'élaboration de drivers d'instruments LabVIEW. Les modèles ont une structure simple et souple et un ensemble commun de VIs de driver d'instrument que vous utilisez pour élaborer votre propre driver. Les modèles établissent un format standard pour tous les drivers LabVIEW et chacun comporte des instructions pour sa modification par rapport à un instrument particulier. Les modèles de drivers d'instruments LabVIEW comportent les onze VIs de composant de modèles suivants :

- “PREFIXE Initialiser” (PREFIX Initialize.vi)
- “PREFIXE Initialiser (VXI, base reg.) (PREFIX Initialize (VXI, Reg-based).vi)
- “PREFIXE Fermer” (PREFIX Close.vi)
- “PREFIXE Réinitialiser” (PREFIX Reset.vi)
- “PREFIXE Auto-test” (PREFIX Self Test.vi)
- “PREFIXE Recherche d'erreur” (PREFIX Error Query.vi)
- “PREFIXE Recherche d'erreur (multiple)” (PREFIX Error Query (Multiple).vi)
- “PREFIXE Message d'erreur” (PREFIX Error Message.vi)
- “PREFIXE Recherche de révision” (PREFIX Revision Query.vi)
- “PREFIXE Modèle basé message” (PREFIX Message-Based Template.vi)
- “PREFIXE Modèle basé registre” (PREFIX Register-Based Template.vi)

Les modèles comprennent les VIs de support suivants :

- “PREFIXE Initialiser le nettoyage de l'utilitaire” (PREFIX Utility Clean Up Initialize.vi)
- “PREFIXE Installer l'instrument par défaut de l'utilitaire” (PREFIX Utility Default Instrument Setup.vi)

Ils comprennent également l'arborescence d'exemple de VI suivant :

- “PREFIXE Arborescence de VI” (PREFIX VI Tree.vi)

Au lieu de développer vos propres VIs pour accomplir ces tâches, utilisez les modèles de VIs de drivers d'instruments LabVIEW, qui sont déjà conformes aux standards de LabVIEW en matière de drivers d'instruments. Les modèles de VIs sont compatibles IEEE 488.2 et fonctionnent avec les instruments IEEE 488.2 avec un minimum de modifications. Pour les instruments autres que les IEEE 488.2, utilisez les modèles de VIs comme squelette ou motif, que vous pouvez modifier en remplaçant ses commandes par les commandes correspondantes spécifiques à l'instrument où il convient. Après avoir modifié les VIs, vous disposez d'un driver de niveau de base qui applique tous les VIs de drivers d'instruments à votre propre instrument.

De plus, les drivers d'instruments LabVIEW qui sont conçus à partir des modèles de VIs ressemblent aux autres drivers d'instruments de la bibliothèque. Vous pouvez ainsi mieux comprendre le fonctionnement des drivers d'instruments lorsque vous les utilisez.

Descriptions des modèles de VIs de drivers d'instruments

Les modèles de VIs de drivers d'instruments suivants sont disponibles.

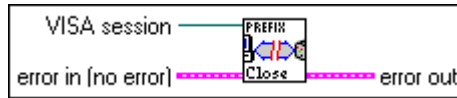


Remarque *Pour développer votre propre VI de driver d'instrument, suivez les instructions contenues sur la face-avant du VI “Modèle” (Template.vi).*

PREFIXE Fermer (PREFIX Close.vi)

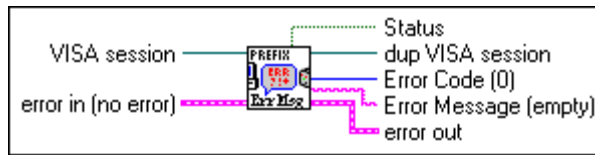
Tous les drivers d'instruments LabVIEW doivent comporter un VI “Fermer” (Close.vi). Ce VI est le dernier à être appelé pendant le contrôle d'un instrument. Il termine la connexion logicielle à l'instrument et désaffecte les ressources du système. De plus, vous pouvez choisir de mettre l'instrument en état de veille. Par exemple, si vous développez un driver

commutateur, vous pouvez déconnecter tous les commutateurs lorsque vous fermez le driver d'instrument.



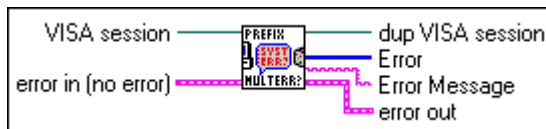
PREFIXE Message d'erreur (PREFIX Error Message.vi)

Le VI "PREFIXE Message d'erreur" (PREFIX Error Message.vi) est un modèle qui sert à créer un VI "Message d'erreur" (Error Message.vi) pour votre instrument particulier. Il traduit les informations sur l'état d'erreur retournées par un VI de driver d'instrument LabVIEW dans une chaîne lisible par l'utilisateur.



PREFIXE Recherche d'erreur (PREFIX Error Query.vi), Recherche d'erreurs (multiples.vi) (Error Query (Multiple).vi) et Message d'erreur (Error Message.vi)

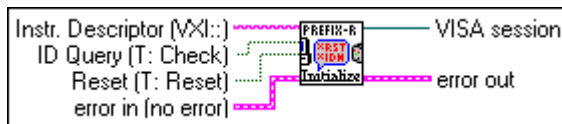
Si un instrument dispose de capacité de recherche d'erreur, le driver d'instrument LabVIEW comporte des VIs "Recherche d'erreur" (Error Query.vi) et "Message d'erreur" (Error Message.vi). Le VI "Recherche d'erreur" (Error Query.vi) interroge l'instrument et retourne des informations sur l'erreur spécifique à l'instrument. Le VI "Message d'erreur" (Error Message.vi) traduit les informations sur l'état d'erreur retournées par un VI de driver d'instrument LabVIEW dans une chaîne lisible par l'utilisateur.



PREFIXE Initialiser (PREFIX Initialize.vi) et PREFIXE Initialiser (VXI, basé registre) (PREFIX Initialize (VXI, Reg-based).vi)

Le VI "Initialiser" (Initialize.vi) est le premier VI appelé quand vous accédez à un driver d'instrument. Il configure l'interface de communications, contrôle les handles et envoie une commande par défaut à l'instrument. En général, la configuration par défaut règle l'opération de l'instrument pour le reste du driver (y compris l'activation et désactivation d'en-têtes, ou l'usage d'interrogations sous forme longue ou courte). Après une opération réussie, le VI "Initialiser" (Initialize.vi) retourne une **session VISA** qui adresse l'instrument dans tous les VIs suivants de driver d'instrument. Le VI "Initialiser" (Initialize.vi) constitue un modèle

pour des instruments basés messages tandis que “Initialiser” (VXI, basé registre) convient aux instruments basés registres.



Le VI comporte une chaîne **Descripteur d’instrument** comme entrée. Selon la syntaxe de cette entrée, le VI configure l’interface d’E/S et génère un handle d’instrument pour tous les autres VIs de driver d’instrument. Le tableau suivant présente la syntaxe du Descripteur d’instrument. Les paramètres en option sont indiqués entre crochets ([]).

Interface	Syntaxe
GPIB	GPIB[carte]::adresse primaire[::adresse secondaire][::INSTR]
VXI	VXI::VXI adresse logique[::INSTR]
GPIB-VXI	GPIB-VXI[carte][::GPIB-VXI adresse primaire]::VXI adresse logique[::INSTR]
Série	ASRL[carte][::INSTR]

Le mot clef GPIB est utilisé avec les instruments GPIB. Le mot clef VXI est utilisé pour des contrôleurs intégrés ou MXIbus. Le mot clef GPIB-VXI est utilisé pour un contrôleur GPIB-VXI de National Instruments.

Le tableau suivant présente les valeurs par défaut pour les paramètres en option.

Paramètre en option	Valeur par défaut
carte	0
adresse secondaire	aucune
GPIB-VXI adresse primaire	1

De plus, le VI “Initialiser” (Initialize.vi) exécute des interrogations ID sélectionnables et des opérations de réinitialisation. Vous pouvez désactiver l’interrogation ID quand vous souhaitez utiliser le driver avec un instrument similaire mais différent sans modifier le code source du driver. Vous pouvez également activer ou désactiver l’opération de réinitialisation. Cette caractéristique est utile lors de la mise au point du driver et vous permet d’éviter que la réinitialisation ne mette l’instrument hors de l’état que vous souhaitez tester.

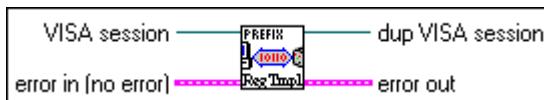
PREFIXE Modèle basé message (PREFIX Message-Based Template.vi) et Modèle basé registre (Register-Based Template.vi)

Les VIs “PREFIXE Modèle basé message” (PREFIX Message-Based Template.vi) et “Modèle basé registre” (Register-Based Template.vi) constituent le point de départ pour développer vos propres VIs de driver d'instrument. Les modèles de VIs comportent toutes les commandes nécessaires au driver d'instrument ainsi que des instructions pour les adapter à un instrument particulier.



PREFIXE Modèle basé registre (PREFIX Register-Based Template.vi)

Le VI “PREFIXE Modèle basé registre” (PREFIX Register-Based Template.vi) est un modèle qui sert à créer un VI basé registre pour votre instrument particulier.



PREFIXE Réinitialiser (PREFIX Reset.vi)

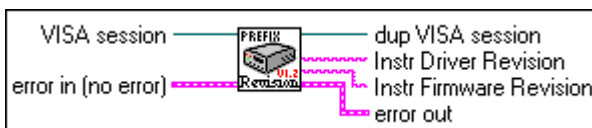
Tous les drivers d'instruments LabVIEW ont un VI “Réinitialiser” (Reset.vi) qui met l'instrument dans un état par défaut. L'état par défaut dans lequel le VI “Réinitialiser” (Reset.vi) met l'instrument est précisé dans les informations d'aide du VI “Réinitialiser” (Reset.vi). Dans le cas d'un instrument IEEE 488.2, ce VI envoie la chaîne de commande *RST à l'instrument. Quand vous réinitialisez l'instrument avec le VI “Initialiser” (Initialize.vi), ce VI est appelé. Vous pouvez également appeler le VI “Réinitialiser” (Reset.vi) séparément. Si l'instrument ne peut pas exécuter la réinitialisation, le VI “Réinitialiser” (Reset.vi) retourne la chaîne `Reset Not Supported` (Réinitialisation non supportée).



PREFIXE Recherche de révision (PREFIX Revision Query.vi)

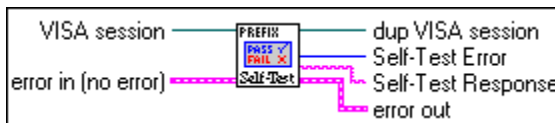
Les drivers d'instruments LabVIEW ont un VI "Recherche de révision" (Revision Query.vi). Ce VI produit les éléments suivants :

- La révision du driver de l'instrument.
- La révision du microprogramme de l'instrument utilisé. Si la révision du microprogramme de l'instrument ne peut pas être interrogée, le VI "Recherche de révision" (Revision Query.vi) retourne la chaîne `Firmware Revision Not Supported` (Révision du microprogramme non supportée).



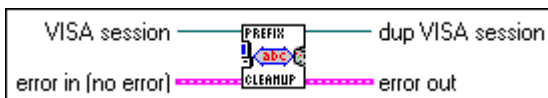
PREFIXE Auto-test (PREFIX Self-Test.vi)

Si un instrument est muni de capacités d'auto-test, le driver d'instrument LabVIEW doit comporter un VI "Auto-test" (Self-Test.vi) qui demande à l'instrument d'effectuer un auto-test et de retourner son résultat. Si l'instrument ne peut pas exécuter l'auto-test, le VI "Auto-test" (Self-Test.vi) retourne la chaîne `Self-Test Not Supported` (Auto-test non supporté).



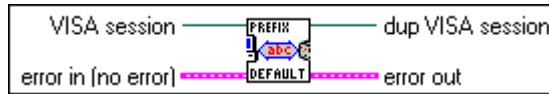
PREFIXE Initialiser le nettoyage de l'utilitaire (PREFIX Utility Clean UP Initialize.vi)

Ferme une session VISA en cours si une erreur survient lors de l'initialisation. Ce VI ne doit être appelé qu'à partir du VI "Initialiser" (Initialize.vi).



PREFIXE Installer l'instrument par défaut de l'utilitaire (PREFIX Utility Default Instrument Setup.vi)

Envoie une chaîne de commande par défaut à l'instrument chaque fois qu'une nouvelle **session VISA** est lancée ou que l'instrument est réinitialisé. Utilisez ce VI comme sous-VI pour les VIs "Initialiser" (Initialize.vi) et "Réinitialiser" (Reset.vi).



PREFIXE Arborescence de VI (PREFIX VI Tree.vi)

Le VI "Arborescence de VI" (VI Tree.vi) est un VI non exécutable qui montre la structure fonctionnelle du driver d'instrument. Il comprend le VI de démarrage, les VIs d'application et les VIs de tous les composants.



La palette VISA contient les sous-palettes suivantes :

- Manipulation d'événements
- Accès de haut niveau aux registres
- Accès de bas niveau aux registres
- Fonctions série

Opérations

Cette section décrit les opérations de référence de la bibliothèque VISA.

Paramètres de référence de la bibliothèque VISA

La plupart des opérations de la bibliothèque VISA utilisent les paramètres suivants :

- Le paramètre **session VISA** est un identificateur logique unique qui sert à communiquer avec une ressource. Cet identificateur est créé et relié à une ressource par la fonction VISA Open. Il est ensuite utilisé par d'autres fonctions VISA pour accéder à la ressource et à ses attributs. La session VISA dupliquée est une copie de la session VISA issue des fonctions VISA. Le fait de transmettre la session VISA à travers les fonctions vous permet de simplifier la programmation du flux de données en enchaînant les fonctions. Ce paramètre est comparable aux numéros de référence de fichiers dupliqués utilisés par les fonctions d'E/S sur fichiers.

La **session VISA** est configurée sur la classe **Instrument** par défaut. Vous pouvez changer le type de classe en ouvrant le menu local sur la commande de session VISA en mode Edition et en sélectionnant une classe différente. Les classes suivantes sont actuellement supportées :

- Instrument
- Instrument GPIB
- Instrument VXI/GPIB-VXI/VME RBD
- Instrument VXI/GPIB-VXI MBD
- Instrument série
- Événement générique
- Événement de requête de service
- Événement de déclenchement
- Événement de signal VXI

- Événement d'interruption VXI/VME
- Gestionnaire de ressource
- Instrument PXI
- MemAcc VXI/GPIB-VXI/VME



Remarque *Les classes Événement générique, Événement de requête de service, Événement de déclenchement, Événement de signal VXI, Événement d'interruption VXI/VME et Gestionnaire de ressources ne peuvent être transmises qu'en tant que session VISA avec la fonction VISA Close et VISA Nœud de propriété.*

Les fonctions VISA n'acceptent pas les mêmes classes de **session VISA**. Les classes valides pour chaque fonction sont indiquées dans la documentation. Par exemple, les fonctions des palettes **Accès de haut niveau aux registres** et **Accès de bas niveau aux registres** n'acceptent pas de sessions VISA de classe Instrument GPIB ou Instrument série. Si vous câblez une **session VISA** à une fonction qui n'accepte pas la classe de la session ou si vous câblez deux sessions VISA de classes différentes, votre diagramme sera brisé et l'erreur sera indiquée en tant que conflit de classe.

- Les terminaux **entrée d'erreur** et **sortie d'erreur** comportent les clusters d'erreur dans chaque fonction VISA. Un cluster d'erreur comprend trois champs. Le champ d'état est un booléen qui est Vrai (TRUE) quand une erreur survient et Faux (FALSE) quand aucune erreur ne survient. Le champ de **code** est une valeur de code d'erreur VISA si une erreur survient lors de l'exécution d'une fonction VISA. L'Annexe A, *Codes d'erreur*, liste les codes d'erreur de la bibliothèque de référence VISA. Le champ **source** est une chaîne de caractères qui décrit où l'erreur est survenue. En câblant la **sortie d'erreur** de chaque fonction à l'**entrée d'erreur** de la fonction suivante, la première condition d'erreur est enregistrée et distribuée à la fin du diagramme, et n'est ainsi indiquée qu'à un seul emplacement.

Descriptions des opérations VISA

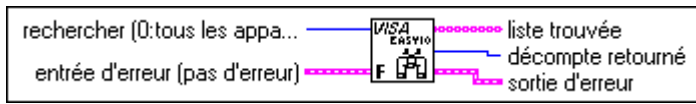
Ces fonctions apparaissent dans la palette principale **VISA**. Les classes valides pour ces fonctions sont Instrument (par défaut), Instrument GPIB, Instrument série, Instrument VXI/GPIB-VXI/VME RBD et Instrument VXI/GPIB-VXI MBD.



Remarque *Les Vis VISA simples suivants constituent une interface simple aux fonctions dont ils se servent. Si votre application nécessite une performance optimale, utilisez les fonctions VISA qui se trouvent également dans cette palette.*

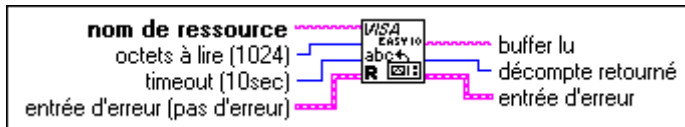
Easy VISA Find Resources

Trouve toutes les ressources VXI, série et GPIB disponibles pour la communication.



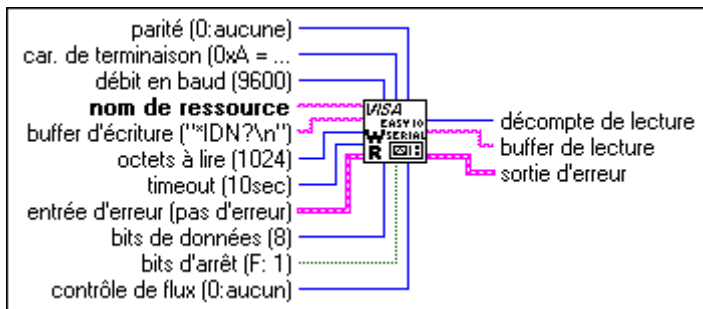
Easy VISA Read

Lit des données de la ressource spécifiée par le nom de la ressource. Le nombre maximal d'octets à lire est déterminé par le décompte d'octets.



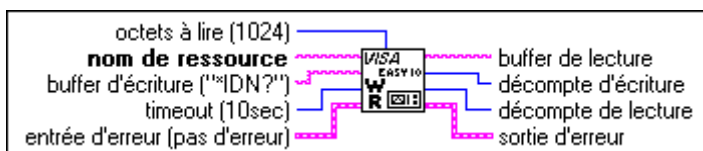
Easy VISA Serial Write and Read

Ecrit une chaîne de commande sur le périphérique série spécifié et lit ensuite la réponse des données. La lecture s'achève après avoir reçu le caractère de terminaison spécifié ou un nombre d'octets spécifiés dans le paramètre de nombre d'octets à lire, le premier des deux prévalant. Si un caractère de terminaison est exigé pour écrire vers l'instrument, il faut l'inclure dans la chaîne de caractères du buffer d'écriture.



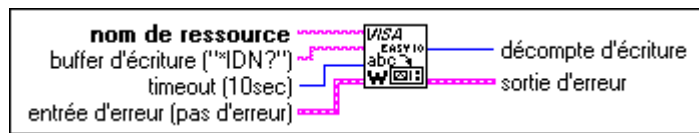
Easy VISA Write and Read

Ecrit une chaîne de commande sur le périphérique spécifié et lit ensuite les données de la réponse.



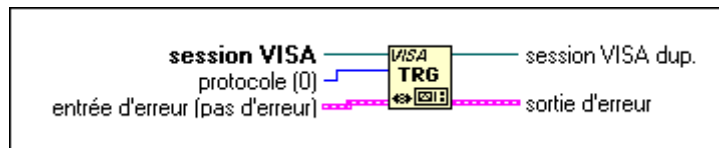
Easy VISA Write

Écrit une chaîne de commande sur le périphérique spécifié.



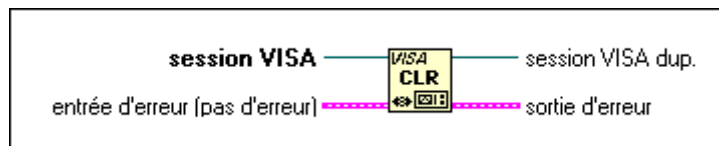
VISA Assert Trigger

Définit un déclenchement logiciel ou matériel, selon le type d'interface.



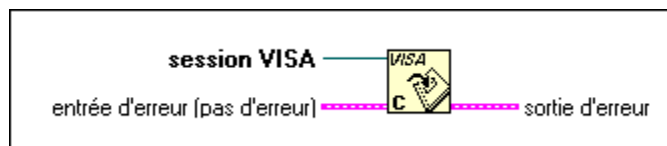
VISA Clear

Exécute une réinitialisation IEEE 488.1 du périphérique. Pour les systèmes VXI, cette fonction est équivalente à “Word Serial Clear” ; pour les systèmes GPIB, c’est la commande “Selected Device Clear”. Pour les systèmes série, la fonction envoie la chaîne de caractères *CLS In.



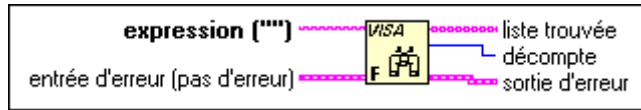
VISA Close

Ferme une session de périphérique spécifié ou un objet d'événement. VISA Close accepte toutes les classes disponibles. Pour une liste des classes disponibles, reportez-vous à la section *Paramètres de référence de la bibliothèque VISA* figurant plus haut dans ce chapitre.



VISA Find Resource

Demande au système de trouver les périphériques associés à une interface spécifiée.



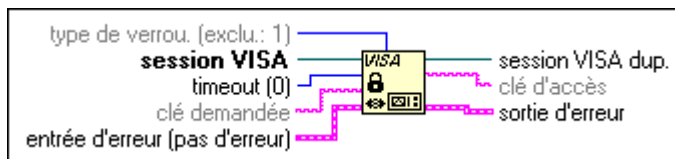
Les tableaux suivants décrivent les paramètres d'expression du VI VISA Find Resource.

Ressources d'instrument	Expression
GPIB	GPIB[0-9]*::?*INSTR
VXI	VXI?*INSTR
GPIB-VXI	GPIB-VXI?*INSTR
GPIB et GPIB-VXI	GPIB?*INSTR
Tous les VXI	?*VXI[0-9]*::?*INSTR
Série	ASRL[0-9]*::?*INSTR
Tous	?*INSTR

Ressources mémoire	Expression
VXI	VXI?*MEMACC
GPIB-VXI	GPIB-VXI?*MEMACC
Tous les VXI	?*VXI[0-9]*?*MEMACC
Tous	?*MEMACC

VISA Lock

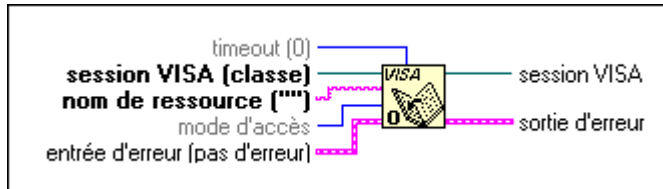
Etablit un accès verrouillé à la ressource spécifiée.



Pour plus d'informations sur le verrouillage et le verrouillage partagé VISA, consultez le chapitre 8, *Tutorial VISA de LabVIEW*, dans le *Manuel de l'utilisateur LabVIEW*.

VISA Open

Ouvre une session sur le périphérique spécifié et retourne un identificateur de session que vous pouvez utiliser pour appeler toute autre opération sur ce périphérique.



Le tableau suivant montre la grammaire des chaînes de caractères de l'adresse. Les segments de chaînes en options figurent entre crochets ([]).

Interface	Syntaxe
VXI	VXI[carte]::adresse logique VXI[::INSTR]
GPIO-VXI	GPIO-VXI[carte]::adresse logique VXI[::INSTR]
GPIO	GPIO[carte]::adresse primaire[::adresse secondaire][::INSTR]
ASRL	ASRL[carte][::INSTR]
VXI	VXI[carte]::MEMACC
GPIO-VXI	GPIO-VXI[carte]::MEMACC

Le mot clef VXI est utilisé pour les instruments VXI par le biais de contrôleurs intégrés ou MXIbus. Le mot clef GPIO-VXI est utilisé pour un contrôleur GPIO-VXI. Le mot clef GPIO peut être utilisé pour établir la communication avec un appareil GPIO. Le mot clef ASRL est utilisé pour établir la communication avec un périphérique série asynchrone (RS-232, par exemple).

Le tableau suivant présente les valeurs par défaut pour les segments de chaînes en option.

Segments de chaînes en option	Valeur par défaut
carte	0
adresse secondaire	aucune

Le tableau suivant présente des exemples de chaînes de caractères d'adresses.

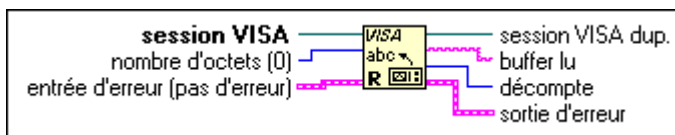
Chaîne de caractères d'adresses	Description
VXI0::1::INSTR	Un périphérique VXI à l'adresse logique 1 dans l'interface VXI VXI0.
GPIB-VXI::9::INSTR	Un périphérique VXI à l'adresse logique 9 dans un système sous contrôle d'un GPIB-VXI.
GPIB::1::0::INSTR	Un appareil GPIB à l'adresse principale 1, à l'adresse secondaire 0 dans une interface GPIB 0 .
ASRL1::INSTR	Un périphérique série connecté à l'interface ASRL1.
VXI::MEMACC	Accès de registres de niveau de carte à l'interface VXI.
GPIB-VXI1::MEMACC	Accès de registres de niveau de carte à l'interface numéro 1 du GPIB-VXI.

Pour le paramètre du mode d'accès, la valeur `VI_EXCLUSIVE_LOCK` (1) est utilisée pour acquérir un verrouillage exclusif dès que la session est ouverte ; si un verrouillage ne peut pas être acquis, la session est fermée et une erreur est retournée.

La valeur `VI_LOAD_CONFIG` (4) est utilisée pour configurer les attributs à des valeurs spécifiées par un utilitaire de configuration externe, comme T&M Explorer (pour Windows 95/NT) ou VISAconf (pour Windows 3.x, Solaris 2 et HP-UX).

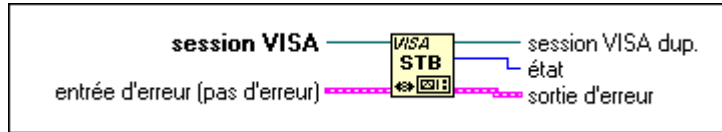
VISA Read

Lit des données d'un périphérique. Les données sont transmises de façon synchrone ou asynchrone en fonction de la plate-forme.



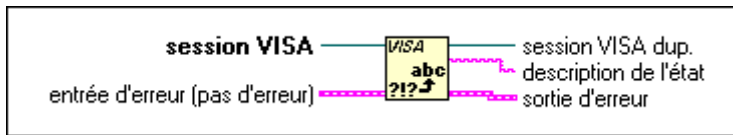
VISA Read STB

Lit l'**état** de la requête de service d'un périphérique basé message. Par exemple, sur l'interface IEEE 488.2, le message est lu par des périphériques d'interrogation. Pour d'autres types d'interfaces, un message est envoyé suite à une requête de service pour récupérer des informations d'état. Si la longueur des informations d'état n'est qu'un octet, l'octet le plus significatif est retourné avec la valeur zéro.



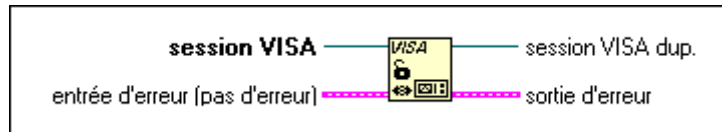
VISA Status Description

Récupère une chaîne de caractères lisible par l'utilisateur qui décrit le code d'état figurant dans l'**entrée d'erreur**.



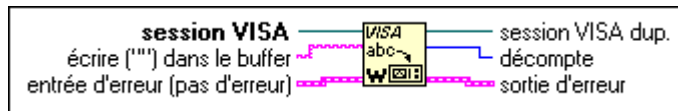
VISA Unlock

Abandonne le verrouillage obtenu précédemment par le biais de la fonction VISA Lock.



VISA Write

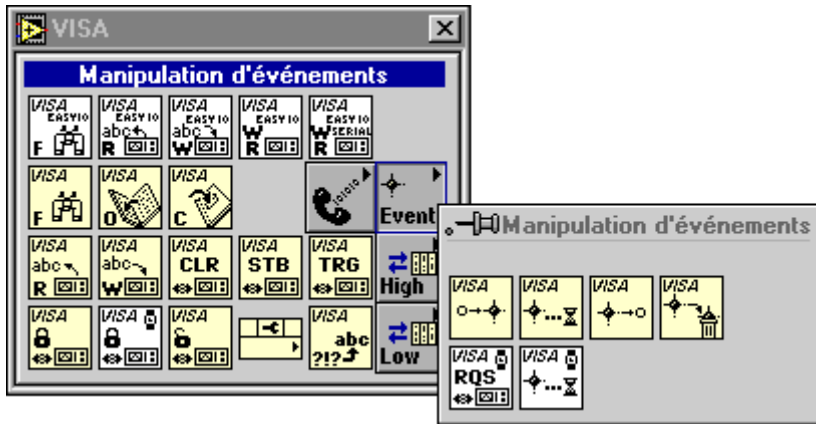
Écrit des données vers le périphérique. Les données sont transmises de façon synchrone ou asynchrone en fonction de la plate-forme.



Fonctions de Gestion d'événements

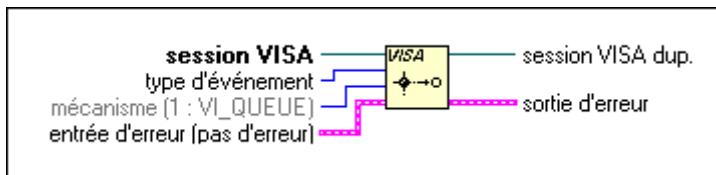
Cette section décrit les fonctions de Gestion d'événements VISA. Les classes valides pour ces fonctions sont Instrument (par défaut), Instrument GPIB, Instrument série, Instrument VXI/GPIB-VXI/VME RBD et Instrument VXI/GPIB-VXI MBD.

Vous pouvez trouver les fonctions de Gestion d'événements VISA dans la palette VISA, accessible en sélectionnant **Fonctions»E/S d'instruments»VISA**.



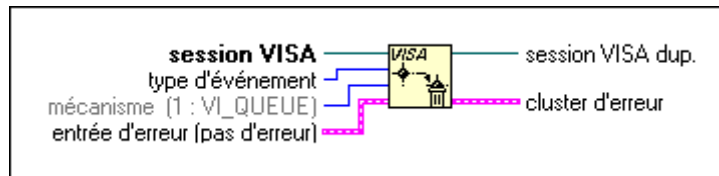
VISA Disable Event

Désactive le traitement d'un événement. Cette opération évite que de nouvelles instances d'un événement soient mises en attente. Cependant, les instances d'événements déjà en file d'attente ne sont pas perdues ; utilisez la fonction VISA Discard Events si vous souhaitez éliminer des événements en file d'attente.



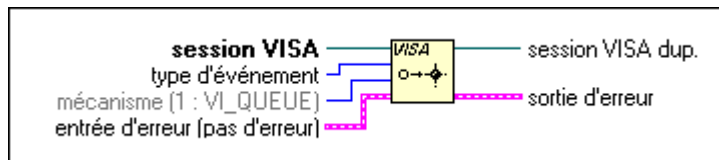
VISA Discard Events

Élimine toutes les instances en attente des types d'événements et des mécanismes spécifiés de la session.



VISA Enable Event

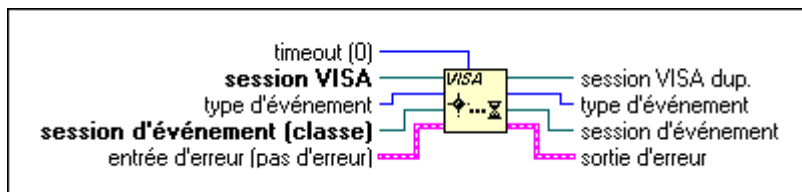
Active la notification d'un événement spécifié.



Remarque Vous devez appeler la fonction *VISA Enable Event* pour une session donnée avant d'utiliser la fonction *VISA Wait on Event*.

VISA Wait On Event

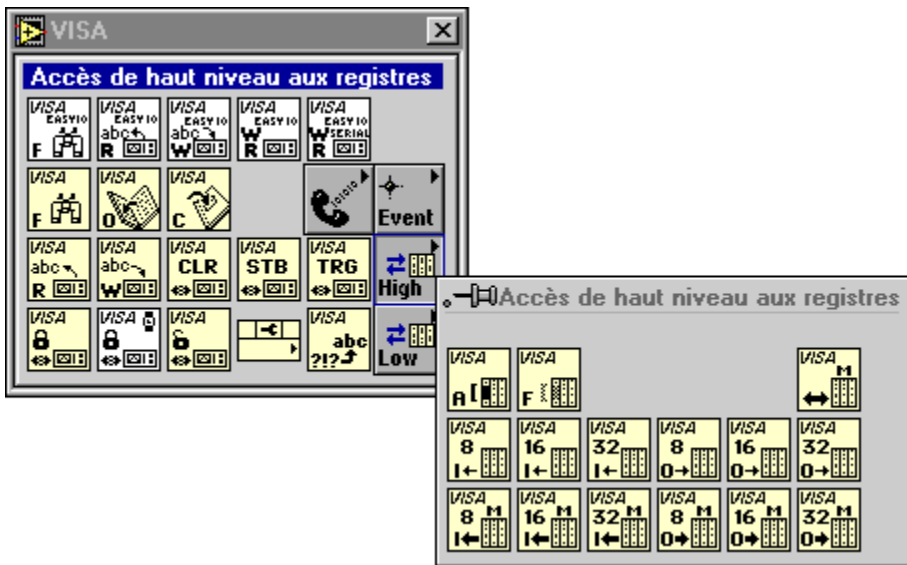
Suspend l'exécution d'un fil d'application et attend un événement de **type d'événement** pour une durée inférieure à **timeout**. Consultez les descriptions d'événements individuels pour obtenir les définitions du contexte. Si le **type d'événement** spécifié est "Tous les événements", l'opération attend n'importe quel événement activé pour cette session.



Remarque Vous devez appeler la fonction *VISA Enable Event* pour une session donnée avant d'utiliser la fonction *VISA Wait on Event*.

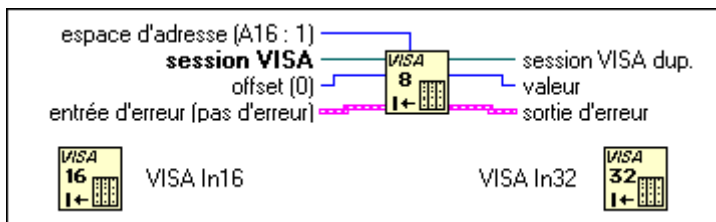
Fonctions d'accès de haut niveau aux registres

Cette section décrit les fonctions VISA d'accès de haut niveau aux registres. Les classes valides pour ces fonctions sont Instrument (par défaut), Instrument GPIB, Instrument série, Instrument VXI/GPIB-VXI/VME RBD et Instrument VXI/GPIB-VXI MBD. Pour accéder aux fonctions VISA d'accès de haut niveau aux registres, ouvrez la sous-palette correspondante de la palette VISA.



VISA In8 / In16 / In32

Lit 8 bits, 16 bits ou 32 bits de données, respectivement, dans l'espace mémoire spécifié (base de mémoire attribuée plus offset). Ces fonctions n'exigent pas d'appeler VISA Map Address avant d'être invoquées.

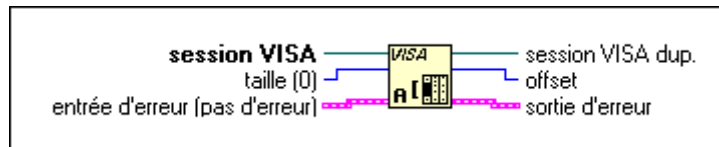


Le tableau suivant liste les entrées valides pour spécifier l'espace d'adresse.

Valeur de l'espace d'adresse	Description
(1)	Adresse l'espace d'adresse A16 du bus VXI/MXI.
(2)	Adresse l'espace d'adresse A24 du bus VXI/MXI.
(3)	Adresse l'espace d'adresse A32 du bus VXI/MXI.

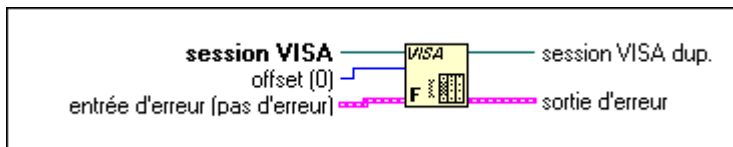
VISA Memory Allocation

Retourne un offset dans la région du périphérique attribué par la session. La mémoire peut être attribuée sur le périphérique même ou sur la mémoire système de l'ordinateur. Si le périphérique auquel la session VISA donnée se rapporte réside sur la carte d'interface locale, la mémoire peut être attribuée sur le périphérique même ou sur la mémoire système de l'ordinateur. La région mémoire référencée par **offset** qui est retournée depuis cette fonction peut être accédée par les fonctions de haut niveau VISA Move In8 / Move In16 / Move In32 et VISA Move out8 / Move out16 / Move out32. Elle peut également être mappée avec la fonction VISA MapAddress.



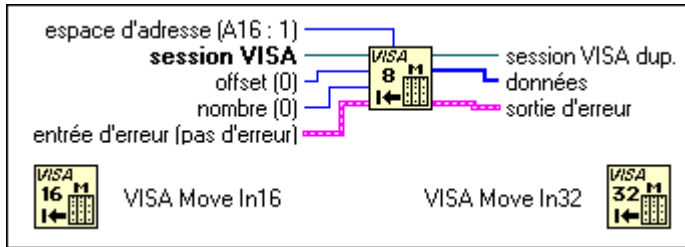
VISA Memory Free

Libère la mémoire précédemment attribuée par la fonction VISA Memory Allocation. Si l'**offset** spécifié a été mappé avec la fonction VISA Map Address, il doit être démappé avant que la mémoire puisse être libérée.



VISA Move In8 / Move In16 / Move In32

Déplace un bloc de données depuis la mémoire périphérique vers la mémoire locale par accès de 8 bits, 16 bits ou 32 bits, respectivement. Les fonctions VISA Move InXX utilisent l'espace d'adresse spécifié pour lire 8 bits, 16 bits ou 32 bits de données, respectivement, depuis l'offset spécifié. Ces fonctions n'exigent pas d'appeler VISA Map Address avant d'être invoquées.

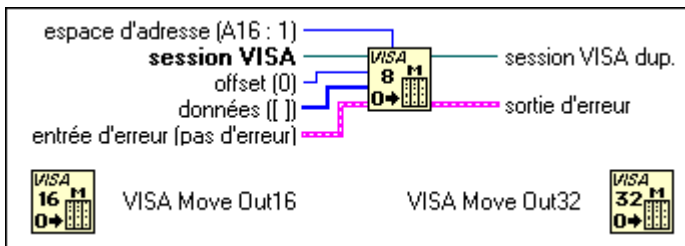


Le tableau suivant liste les entrées valides pour spécifier l'espace d'adresse.

Valeur	Description
(1)	Adresse l'espace d'adresse A16 du bus VXI/MXI.
(2)	Adresse l'espace d'adresse A24 du bus VXI/MXI.
(3)	Adresse l'espace d'adresse A32 du bus VXI/MXI.

VISA Move Out8 / Move Out16 / Move Out32

Déplace un bloc de données depuis la mémoire locale vers l'adresse et l'offset spécifiés et utilise l'espace d'adresse spécifié pour écrire 8 bits, 16 bits ou 32 bits de données, respectivement, à l'offset spécifié. Ces fonctions n'exigent pas d'appeler VISA Map Address avant d'être invoquées.

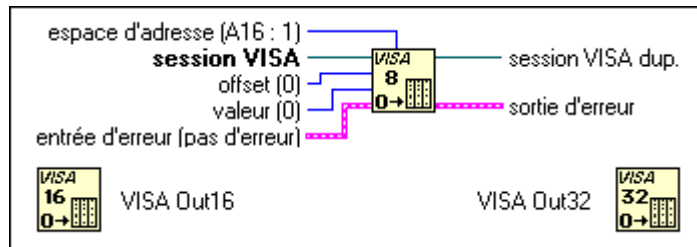


Le tableau suivant liste les entrées valides pour spécifier l'espace d'adresse.

Valeur	Description
(1)	Adresse l'espace d'adresse A16 du bus VXI/MXI.
(2)	Adresse l'espace d'adresse A24 du bus VXI/MXI.
(3)	Adresse l'espace d'adresse A32 du bus VXI/MXI.

VISA Out8 / Out16 / Out32

Ecrit 8 bits, 16 bits ou 32 bits de données, respectivement, à partir de la mémoire spécifiée (base de mémoire attribuée plus offset). Ces fonctions n'exigent pas d'appeler VISA Map Address avant d'être invoquées.

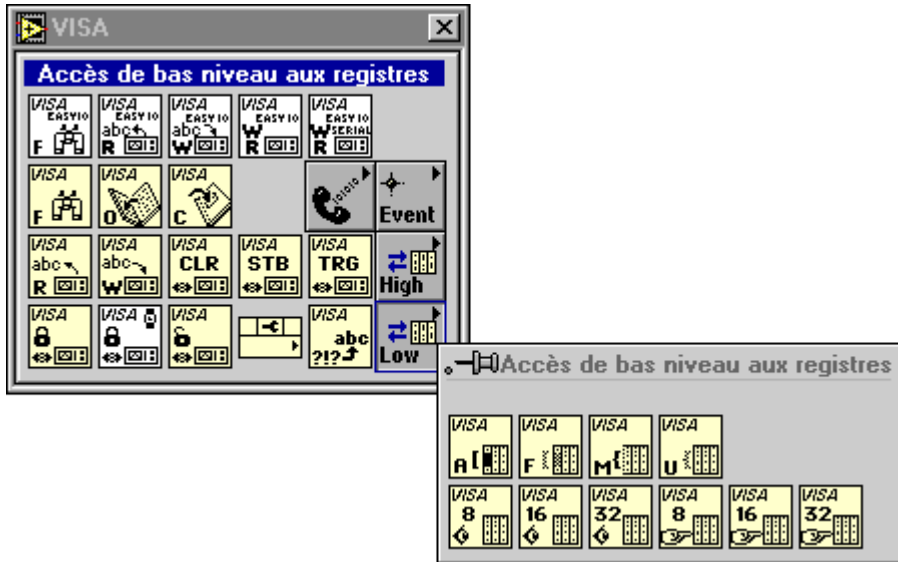


Le tableau suivant liste les entrées valides pour spécifier l'espace d'adresse.

Valeur	Description
(1)	Adresse l'espace d'adresse A16 du bus VXI/MXI.
(2)	Adresse l'espace d'adresse A24 du bus VXI/MXI.
(3)	Adresse l'espace d'adresse A32 du bus VXI/MXI.

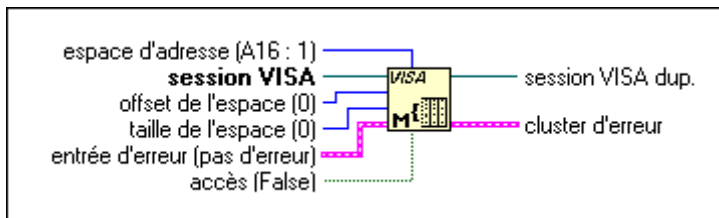
Fonctions d'accès de bas niveau aux registres

Cette section décrit les fonctions VISA d'accès de bas niveau aux registres. Les classes valides pour ces fonctions sont Instrument (par défaut), Instrument GPIB, Instrument série, Instrument VXI/GPIB-VXI/VME RBD et Instrument VXI/GPIB-VXI MBD. Pour accéder aux fonctions VISA d'accès de bas niveau aux registres, ouvrez la sous-palette correspondante de la palette VISA.



VISA Map Address

Mappe un espace de mémoire spécifié. L'espace de mémoire mappé dépend du type d'interface spécifié par **session VISA** et du paramètre **espace d'adresse**. Une fois la fenêtre mappée, VISA repère la fenêtre mappée. Ce comportement fait que VISA ne peut mapper qu'une seule fenêtre par session VISA.



Le tableau suivant liste les entrées valides pour spécifier l'espace d'adresse.

Valeur	Description
(1)	Adresse l'espace d'adresse A16 du bus VXI/MXI.
(2)	Adresse l'espace d'adresse A24 du bus VXI/MXI.
(3)	Adresse l'espace d'adresse A32 du bus VXI/MXI.

VISA Memory Allocation

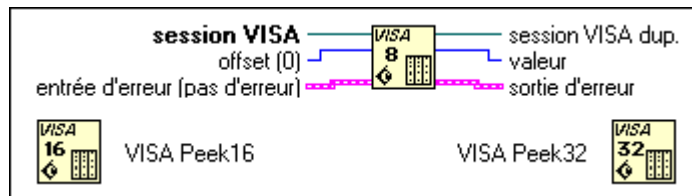
Pour plus d'informations sur la fonction VISA Memory Allocation, reportez-vous à la section *Fonctions d'accès de haut niveau aux registres* de ce chapitre.

VISA Memory Free

Pour plus d'informations sur la fonction VISA Memory Free, reportez-vous à la section *Fonctions d'accès de haut niveau aux registres* de ce chapitre.

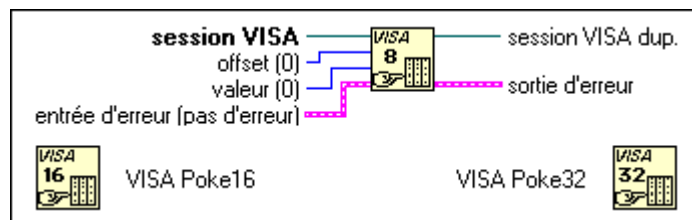
VISA Peek8 / Peek16 / Peek32

Lit une valeur de 8 bits, 16 bits ou 32 bits, respectivement, à partir de l'emplacement d'adresse spécifié par **offset**. L'adresse doit être une adresse de mémoire valide dans le processus actuellement mappé par un appel antérieur de la fonction VISA Map Address.



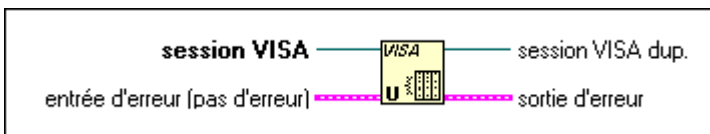
VISA Poke8 / Poke16 / Poke32

Ecrit une valeur de 8 bits, 16 bits ou 32 bits, respectivement, à l'adresse spécifiée et stocke le contenu de la valeur à l'adresse indiquée par **offset**. L'adresse doit être une adresse de mémoire valide dans le processus actuellement mappé par un appel antérieur de la fonction VISA Map Address.



VISA Unmap Address

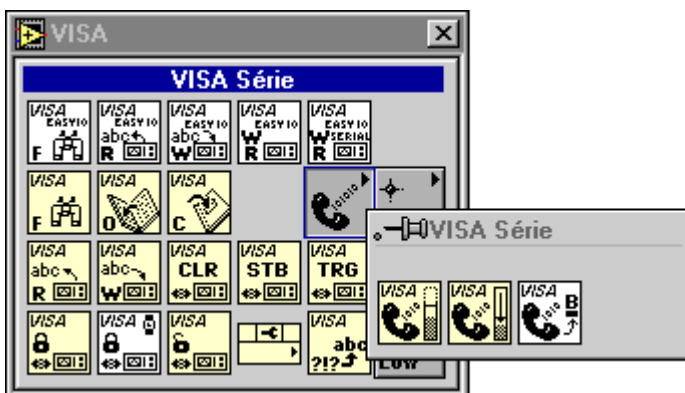
Démappe l'espace de mémoire mappée antérieurement par VISA Map Address.



Fonctions VISA série

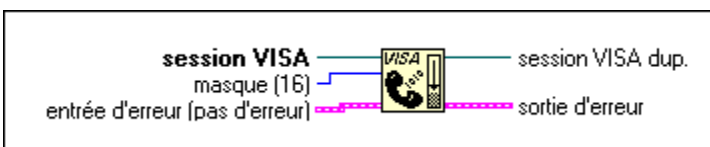
Cette section décrit les fonctions VISA spécifiques aux ports série. Les classes valides pour ces fonctions sont Instrument (par défaut) et Instrument série.

Pour accéder aux fonctions VISA série, ouvrez la sous-palette correspondante de la palette VISA.



VISA Flush Serial Buffer

Vide le buffer série.

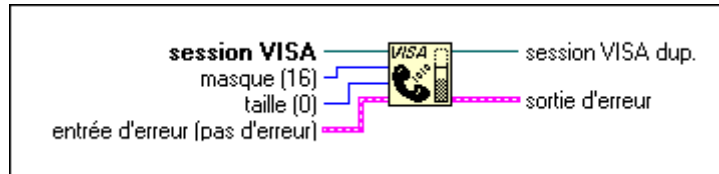


Vider le buffer de réception (16) élimine son contenu, tandis que vider le buffer de sortie (32) attend que le contenu restant du buffer de transmission soit envoyé au périphérique. Pour éliminer toute donnée restante du buffer de transmission, vous devez utiliser le masque

d'élimination du buffer de sortie (128). Pour vider plus d'un buffer en même temps, combinez les masques des buffers en utilisant un OU logique.

VISA Set Serial Buffer Size

Etablit la taille du buffer série.

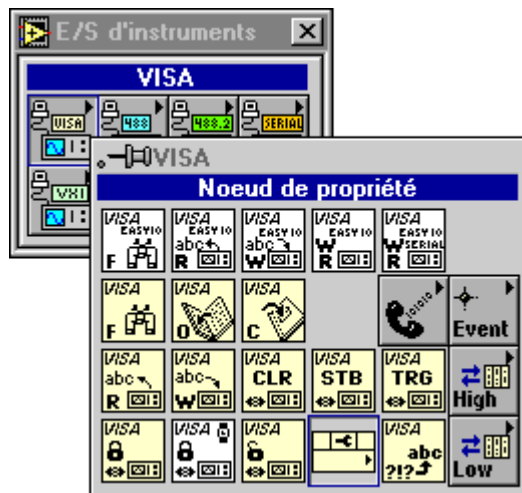


Les valeurs valides pour **masque** sont Buffer de réception série (16) et Buffer de transfert série (32). Pour régler la taille des deux buffers en même temps, combinez les masques des buffers en utilisant un OU logique.

Nœud de propriété VISA

Cette section décrit les attributs de la bibliothèque VISA. Le Nœud de propriété VISA obtient et/ou établit les attributs indiqués. Le nœud est expansible ; l'évaluation commence depuis le haut et progresse vers le bas jusqu'à ce qu'une erreur ou l'évaluation finale survienne.

Pour accéder à ce nœud, sélectionnez **Fonctions»E/S d'instruments»VISA**. Sélectionnez ensuite l'icône Nœud de propriété située dans la rangée du bas de la palette **VISA**.



Le Nœud de propriété VISA n'affiche que les attributs pour la classe de la session qui lui est câblée. Vous pouvez changer la classe d'un Nœud de propriété VISA si vous ne l'avez pas câblé à une **session VISA**. Une fois qu'une **session VISA** est câblée à un Nœud de propriété VISA, elle s'adapte à la classe de la session et tout attribut non valable pour cette classe devient invalide (ce qui est indiqué lorsque l'élément d'attributs devient noir).

Vu que le Nœud de propriété peut s'utiliser dans d'autres contextes dans LabVIEW, il est possible que sa classe par défaut soit autre que la classe VISA lorsque vous le déposez sur le diagramme. Quand vous le câblez à une **session VISA**, il devient une classe VISA.

Descriptions de Nœud de propriété VISA

Les catégories de Nœud de propriété VISA suivantes sont disponibles.

Voie de données rapides

Spécifie les informations suivantes :

- Numéro de voie
- Transmissions de données à travers des paires de voies
- Signal d'activation
- Transmissions en mode normal ou en mode de communication

Paramètres généraux

Déterminent les propriétés suivantes :

- Longueur maximale de la file d'attente de l'événement
- Nom de ressource VISA unique
- Etat de verrouillage de la ressource
- Valeur de timeout pour accéder au périphérique
- Mécanisme de déclenchement de communication
- Informations utilisées pour documenter la fonctionnalité dans votre application VISA

Paramètres GPIB

Spécifient les informations suivantes :

- Les adresses primaire et secondaire d'un appareil GPIB
- Si l'appareil GPIB nécessite un réadressage avant chaque transfert
- Si l'appareil GPIB n'est pas adressé après chaque opération de lecture et d'écriture

Informations d'interface

Fournit des informations sur le type d'interface VISA, le numéro de carte de l'interface et le numéro de carte du périphérique parent.

Paramètres basés messages

Déterminent les aspects suivants de la communication VISA basée messages :

- Protocole d'E/S (GPIB, série, VXI)
- Si un indicateur END est envoyé lors des opérations de lecture
- Si un indicateur END est ignoré lors des opérations de lecture
- Si les opérations de lecture doivent s'achever par un caractère spécial

Paramètres de la ligne modem

Déterminent l'état courant des signaux suivants utilisés lors des communications par modem : CTS, DCD, DSR, DTR, RI et RTS.

Ressources PXI

Spécifient le type d'adresse, la base d'adresse et la taille de l'adresse des périphériques situés dans les emplacements BAR0 à BAR5.

Paramètres PXI

Spécifient les informations suivantes : numéro de périphérique, numéro de fonction, renseignements relatifs au fabricant du sous-système et code du modèle de sous-système.

Paramètres basés registres

Déterminent les aspects suivants de la communication VISA basée registres :

- Identification du fabricant du périphérique
- Modèle du périphérique
- Emplacement physique du périphérique
- Nombre d'éléments dans des opérations de déplacement en bloc aux adresses de source et de mémoire
- **(Windows)** Adresse de base, taille et accès à l'espace

Paramètres série

Spécifient les valeurs suivantes : nombre d'octets au port série, débit en bauds, bits de données, parité, bits d'arrêt, contrôle du flux et méthode de terminaison pour les opérations de lecture et d'écriture.

Informations sur la version

Fournit des informations relatives à la version et au nom du fabricant de l'implémentation VISA, ainsi que la version de la spécification VISA.

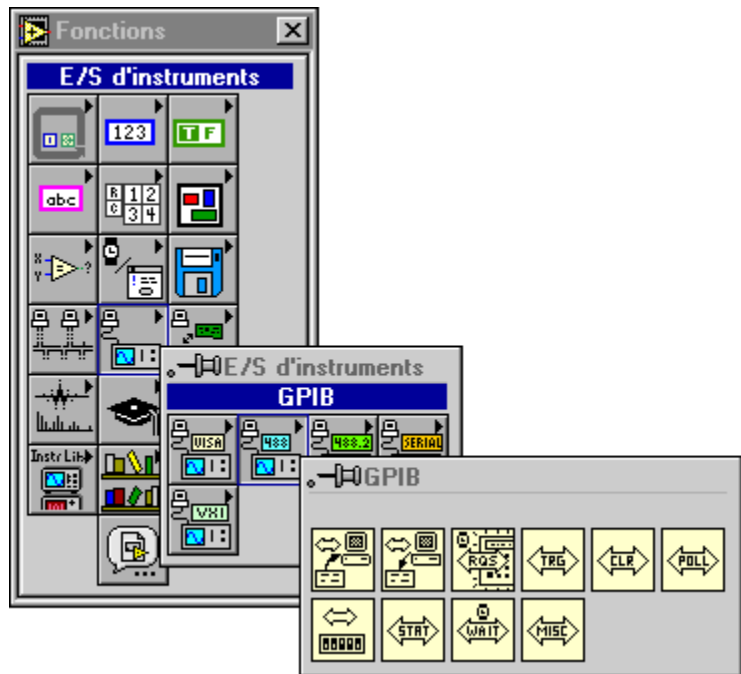
Paramètres VME/VXE

Déterminent les adresses nécessaires, les privilèges d'accès, l'espace de mémoire et les ordres d'octets nécessaires aux communications VXI.

Fonctions GPIB traditionnelles

Ce chapitre décrit les fonctions GPIB traditionnelles.

La figure ci-après montre la palette des fonctions GPIB traditionnelles que vous lancez en sélectionnant **Fonctions»E/S d'instruments»GPIB**.



Pour obtenir des exemples sur l'utilisation des fonctions GPIB traditionnelles, reportez-vous à la bibliothèque `examples\instr\smpgpiib.llb`.

Paramètres des fonctions GPIB traditionnelles

La plupart des fonctions GPIB traditionnelles utilisent les paramètres suivants :

- **adresse** comprend l'adresse de l'appareil GPIB avec lequel la fonction communique. Vous pouvez entrer des adresses primaires et secondaires dans **adresse** en utilisant la forme *primaire+secondaire*. Les valeurs *primaire et secondaire sont décimales* ; ainsi, si *primaire est 2 et secondaire est 3*, **adresse** est 2+3.

Si vous ne spécifiez pas d'adresse, les fonctions n'effectuent pas d'adressage avant de lire et d'écrire la chaîne de caractères. Elles supposent que vous avez envoyé ces commandes par d'autres moyens ou qu'un autre contrôleur est en charge et exécute l'adressage. Si le contrôleur est censé adresser l'appareil mais ne le fait pas avant la fin d'un délai temporel, les fonctions se terminent par une erreur GPIB 6 (timeout) et activent le bit 14 dans la sortie **état**. Si le GPIB n'est pas le Contrôleur en charge, ne spécifiez pas d'**adresse**.

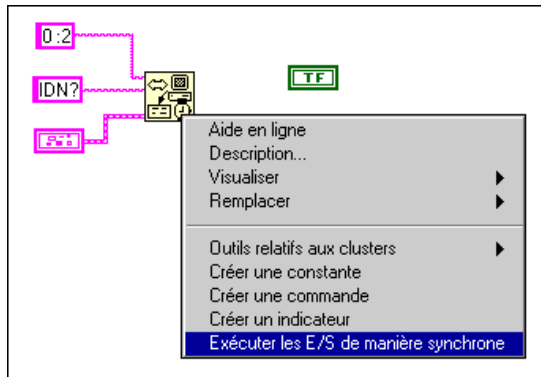
Lorsqu'il existe plusieurs contrôleurs GPIB que LabVIEW peut utiliser, un préfixe pour **adresse** de la forme de ID:adresse (ou ID: si aucune adresse n'est nécessaire) détermine le contrôleur qu'une fonction spécifique utilise. Si aucune identification du contrôleur n'est présente, les fonctions prennent le contrôleur (ou bus) 0.

- **état** est un tableau booléen de 16 bits où chaque bit décrit un état du contrôleur GPIB. Si une erreur survient, le bit 15 est activé. Le champ **code d'erreur** du cluster **sortie d'erreur** contient un code d'erreur GPIB seulement si le bit 15 d'**état** est activé.
- Les terminaux **entrée d'erreur** et **sortie d'erreur** constituent les clusters d'erreur de chaque fonction GPIB traditionnelle. Le cluster d'erreur comprend trois champs. Le champ **état** est un booléen qui est TRUE quand une erreur survient, FALSE quand il n'y a pas d'erreur. Le champ **code** est une valeur du code d'erreur GPIB si une erreur survient lors de l'exécution d'une fonction GPIB. Le champ **source** est une chaîne de caractères qui décrit où l'erreur est survenue. Si le champ **état** du paramètre **entrée d'erreur** d'une fonction est activé, la fonction n'est pas exécutée et le même cluster d'erreur est transmis à la **sortie d'erreur**. En câblant la **sortie d'erreur** de chaque fonction à l'**entrée d'erreur** de la prochaine fonction, la première condition d'erreur est enregistrée et distribuée à la fin du diagramme où elle est indiquée à un seul emplacement.

Comportement des fonctions GPIB traditionnelles

Les fonctions GPIB Read et GPIB Write laissent l'appareil en état adressé lorsque leur exécution s'achève. Si votre appareil ne peut pas fonctionner en état adressé, utilisez la fonction GPIB Misc pour envoyer le message approprié de déadressage ou configurez le logiciel NI-488.2 pour désadresser automatiquement tous les appareils sur le GPIB.

Les fonctions traditionnelles GPIB Read et GPIB Write peuvent s'exécuter en mode asynchrone. Ceci signifie que d'autres activités de LabVIEW peuvent continuer pendant l'opération de ces fonctions GPIB. Lorsque ces fonctions sont configurées en exécution asynchrone, une petite montre apparaît sur les icônes des fonctions. Un élément du menu local des fonctions GPIB Read et GPIB Write vous permet d'activer ou de désactiver le mode asynchrone.

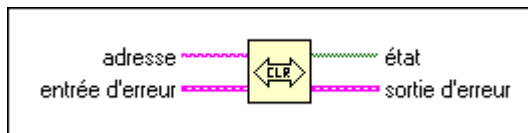


Descriptions des fonctions GPIB traditionnelles

Les fonctions GPIB traditionnelles suivantes sont disponibles.

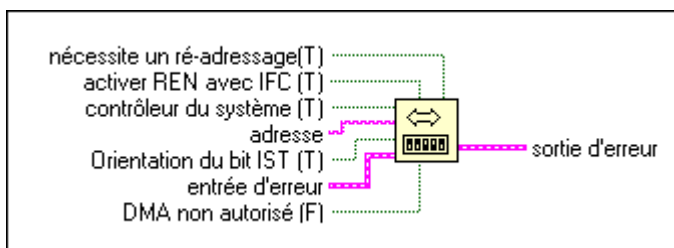
GPIB Clear

Transmet soit "Réinitialisation de l'appareil sélectionné" (SDC) ou "Réinitialisation de l'appareil" (DCL).



GPIB Initialization

Configure l'interface GPIB à l'adresse spécifiée.



GPIB Misc

Exécute l'opération GPIB indiquée par **commande**. Utilisez cette fonction de bas niveau lorsque les fonctions de haut niveau décrites plus haut ne sont pas appropriées.

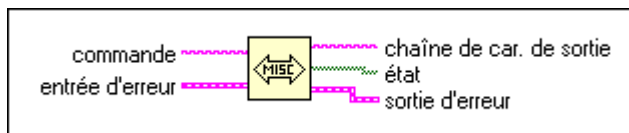


Tableau 34-1. Chaînes de commande des fonctions d'appareil

Fonctions d'appareil	Description
adresse loc	Passer en local.
adresse off	Mettre l'appareil hors ligne.
adresse pct	Transmettre le contrôle.
adresse d'octet ppc	Configurer l'interrogation parallèle (parallel poll) (activer ou désactiver).

Tableau 34-2. Chaînes de commande des fonctions du contrôleur

Fonctions du contrôleur	Description
cac 0/1	Devenir le contrôleur actif.
cmd string	Envoyer des commandes IEEE 488.
dma 0/1	Etablir le mode DMA ou le mode E/S programmées.
gts 0/1	Passer du contrôleur actif à l'état de veille.

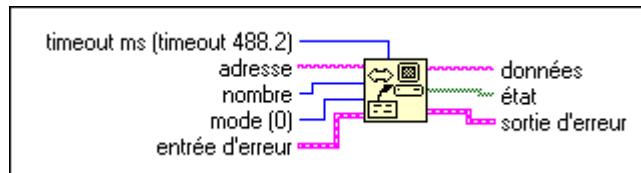
Tableau 34-2. Chaînes de commande des fonctions du contrôleur (Suite)

Fonctions du contrôleur	Description
ist 0/1	Définir le bit d'état individuel.
llo	Verrouillage local.
loc	Mettre le contrôleur en état local.
off	Mettre le contrôleur hors ligne.
ppc octet	Configurer l'interrogation parallèle (parallel poll) (activer ou désactiver).
ppu	Déconfigurer l'interrogation parallèle (parallel poll) pour tous les appareils.
rpp	Effectuer une interrogation parallèle (parallel poll).
rsc 0/1	Demander ou affranchir le contrôle du système.
rsv octet	Demander un service et/ou activer l'octet d'état de l'interrogation en série.
sic	Envoyer une réinitialisation d'interface et établir une activation à distance.
sre 0/1	Etablir ou réinitialiser l'activation à distance.

Pour spécifier le contrôleur GPIB utilisé par cette fonction, utilisez une **commande** sous la forme ID: xxx, où ID est le contrôleur GPIB (numéro de bus) et xxx est la commande de trois caractères ainsi que ses arguments correspondants, si nécessaire. Si vous ne spécifiez pas d'ID du contrôleur, LabVIEW suppose que c'est 0.

GPIB Read

Lit **nombre d'octets** sur l'appareil GPIB situé à **adresse**.

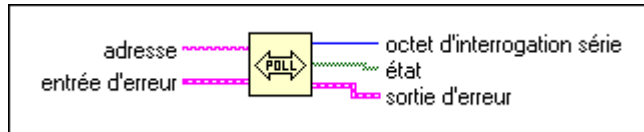


Vous devez utiliser la fonction SetTimeOut pour changer la valeur par défaut (le timeout 488.2 global) de **timeout ms**. Au début, la valeur par défaut de **timeout ms** est 10,000. Reportez-

vous à la description de la fonction SetTimeOut fonction au chapitre 35, *Fonctions GPIB 488.2*, pour plus d'informations.

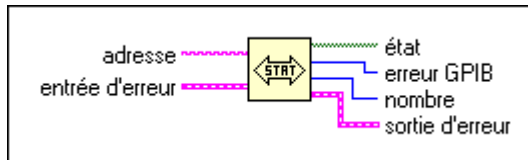
GPIB Serial Poll

Effectue une interrogation en série (serial poll) de l'appareil indiqué par **adresse**.



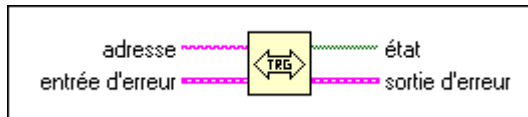
GPIB Status

Détermine l'état du contrôleur GPIB indiqué par **adresse** après l'opération GPIB précédente.



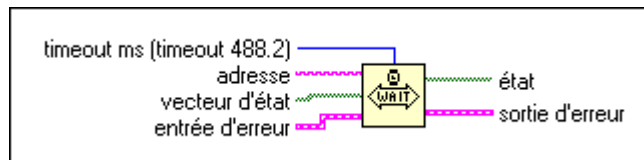
GPIB Trigger

Transmet GET (Group Execute Trigger [déclenchement d'exécution en groupe]) à l'appareil spécifié par **adresse**.



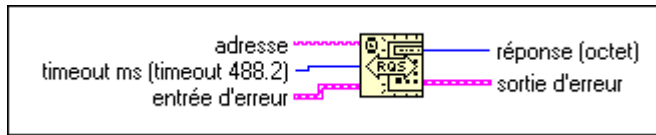
GPIB Wait

Attend les états indiqués par **vecteur d'état** de l'appareil spécifié par **adresse**.



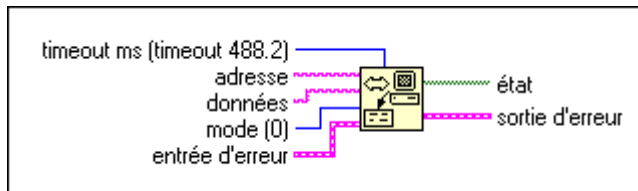
Wait for GPIB RQS

Attend que l'appareil indiqué par **adresse** active la ligne RQS.



GPIB Write

Ecrit les **données** à l'appareil GPIB identifié par **adresse**.



Fonctions d'appareil et de contrôleur GPIB

Cette section décrit les fonctions figurant dans la description de la fonction GPIB Misc. Les fonctions d'appareils envoient des informations de configuration à un instrument spécifique (appareil). Les fonctions de contrôleur GPIB configurent le contrôleur ou envoient des commandes IEEE 488 auxquelles tous les instruments répondent. Observez qu'il existe des versions appareil et contrôleur des commandes **ppc** et **loc**. La syntaxe et l'usage des commandes sont légèrement différents pour chaque version.

Vous pouvez utiliser ces fonctions avec tous les contrôleurs GPIB accessibles par LabVIEW, sauf quand les descriptions des fonctions suivantes indiquent le contraire. Une erreur ECMD (17) survient quand vous exécutez une fonction pour un contrôleur GPIB qui ne dispose pas de la capacité spécifiée. La syntaxe de la fonction est stricte. Chaque fonction ne reconnaît que les minuscules et ne permet qu'un seul espace entre le nom de la fonction et les arguments.

Fonctions d'appareil

loc – Passer en local

syntaxe **loc** *adresse*

loc change temporairement des appareils d'un mode de programmation à distance dans un mode local.

adresse est l'adresse GPIB de l'appareil. Cet argument indique les adresses primaire et secondaire si vous utilisez la forme *primaire + secondaire*, où *primaire et secondaire* sont les valeurs décimales des adresses primaire et secondaire. Par exemple, si *primaire est 2* et *secondaire est 3*, alors l'*adresse* est 2 + 3.

loc transmet le message "Atteindre local" (GTL [Go to local]) à l'appareil GPIB.

off – Mettre l'appareil hors ligne

syntaxe **off** *adresse*

off met l'appareil à l'adresse spécifiée GPIB hors ligne. Ceci n'est nécessaire que lorsqu'un appareil est partagé avec une autre application qui utilise la bibliothèque NI-488 GPIB.

adresse est l'adresse GPIB de l'appareil. Cet argument indique les adresses primaire et secondaire si vous utilisez la forme *primaire + secondaire*, où *primaire et secondaire* sont les valeurs décimales des adresses primaire et secondaire. Par exemple, si *primaire est 2* et *secondaire est 3*, alors l'*adresse* est 2 + 3.

pct – Transmettre le contrôle

syntaxe **pct** *adresse*

pct passe l'autorité de contrôleur en charge (CIC) (Contrôleur en charge) à l'appareil à l'adresse spécifiée. Le contrôleur GPIB passe automatiquement à l'état de veille. La fonction suppose que l'appareil à qui **pct** passe le contrôle détient une capacité de Contrôleur.

adresse est l'adresse GPIB de l'appareil. Cet argument indique les adresses primaire et secondaire si vous utilisez la forme *primaire + secondaire*, où *primaire et secondaire* sont les valeurs décimales des adresses primaire et secondaire. Par exemple, si *primaire est 2* et *secondaire est 3*, alors l'*adresse* est 2 + 3.

pct transmet la séquence de commande suivante :

1. Adresse “Talk” de l’appareil
2. Adresse secondaire de l’appareil, si applicable
3. Prendre le contrôle (TCT [Take Control])

ppc – Configurer l’interrogation parallèle (parallel poll)

syntaxe **ppc** *octet adresse*

ppc permet à l’instrument de répondre à des interrogations parallèles (parallel poll).

octet est 0 ou une commande d’interrogation parallèle valide (PPE). Si *octet* est 0, l’octet de désactivation d’interrogation parallèle (PPD) 0x70 est envoyé pour désactiver l’appareil quand une interrogation parallèle survient. Chacun des 16 messages (PPE) sélectionne une ligne de données GPIB (DIO1 à DIO8) et un sens (1 ou 0) que l’appareil doit utiliser quand il répond au message “Identifier” (IDY [Identify]) lors d’une interrogation parallèle. L’appareil compare le sens **ist** et dirige la ligne indiquée DIO sur TRUE ou FALSE.

adresse est l’adresse GPIB de l’appareil. Cet argument indique les adresses primaire et secondaire si vous utilisez la forme *primaire + secondaire*, où *primaire et secondaire* sont les valeurs décimales des adresses primaire et secondaire. Par exemple, si *primaire est 2 et secondaire est 3*, alors l’*adresse* est 2 + 3.

Fonctions de contrôleur

cac – Devenir le contrôleur actif

syntaxe **cac** 0 (prendre le contrôle de façon synchrone)

cac 1 (prendre le contrôle immédiatement)

cac prend le contrôle de façon synchrone ou immédiatement (et parfois de façon asynchrone). En général, la fonction **cac** n’est pas nécessaire car d’autres fonctions, telles que **cmd** et **rpp**, prennent le contrôle automatiquement.

Si vous essayez de prendre le contrôle de façon synchrone lorsqu’un handshake de données est en cours, la fonction diffère la prise de contrôle jusqu’à ce que le handshake s’achève. Si un handshake n’est pas en cours,

la fonction exécute immédiatement la prise de contrôle. La prise de contrôle de façon synchrone n'est pas garantie si une opération de lecture ou d'écriture s'achève avec un timeout ou toute autre erreur.

Effectuez la prise de contrôle de façon asynchrone quand il est impossible d'obtenir le contrôle de façon synchrone (par exemple, après une erreur de timeout).

L'erreur ECIC survient si le contrôleur GPIB n'est pas le contrôleur en charge (CIC).

cmd – Envoyer des commandes IEEE 488

syntaxe **cmd** chaîne de caractères

cmd transmet les messages de commande GPIB. Ces messages de commande comprennent les adresses “Talk” (parler) et “Listen” (écouter) de l'appareil, les messages de configuration des interrogations série et parallèle et les messages de réinitialisation et de déclenchement de l'appareil.

Vous n'utilisez pas **cmd** pour transmettre des instructions de programmation aux appareils. Les fonctions GPIB Read et GPIB Write transmettent des instructions de programmation et d'autres informations qui dépendent de l'appareil.

chaîne de caractères comprend les octets de commande que le contrôleur transmet. Les caractères ASCII représentent ces octets dans la *chaîne de caractères cmd*. Si vous devez envoyer des caractères non affichables, vous pouvez activer les codes des barres obliques inverses sur la commande de chaîne de caractères ou la constante de la chaîne de caractères, ou vous pouvez utiliser une fonction de formatage pour lister les commandes au format hexadécimal.

dma – Etablir le mode DMA ou le mode d'E/S programmées

syntaxe **dma** 0 (utilisez les E/S programmées)

dma 1 (utilisez les DMA)

dma indique si les transferts de données utilisent le mode DMA.

Quelques cartes GPIB ne sont pas munies de capacité DMA. Si vous essayez d'exécuter **dma** 1, la fonction retourne une erreur GPIB 11 pour indiquer une capacité non disponible.

gts – Passer du contrôleur actif à l'état de veille

syntaxe **gts** 0 (pas de handshaking “shadow”)

gts 1 (handshaking “shadow”)

Description :

gts règle le contrôleur GPIB sur l'état de contrôleur de veille et désactive l'assertion du signal ATN si c'est le contrôleur actif. En général, le contrôleur GPIB prend part au transfert de données. La commande **gts** permet aux appareils GPIB de transférer des données sans la participation du contrôleur GPIB.

Si le handshaking “shadow” est activé, le contrôleur GPIB participe au transfert GPIB en tant que “Listener”, mais n'accepte aucune donnée. Quand il détecte le message END (fin), le contrôleur GPIB active la ligne NFRD (Not Ready For Data [n'est pas prêt pour des données]) pour créer un état de suspension du handshake.

Si le handshaking “shadow” n'est pas activé, le contrôleur GPIB n'effectue ni le handshaking “shadow” ni la suspension du handshake.

Si vous activez l'option du handshake “shadow”, le contrôleur GPIB participe au handshake de données en tant que “Listener” sans lire les données. Il surveille le transfert pour repérer le message END et arrête les transferts suivants. Ce mécanisme permet au contrôleur GPIB de prendre le contrôle de façon synchrone lors d'opérations ultérieures telles que **cmd** ou **rpp**.

Après avoir envoyé la commande **gts**, vous devez toujours attendre END avant de lancer une autre commande GPIB. Vous pouvez le faire avec la fonction GPIB Wait.

L'erreur ECIC survient si le contrôleur GPIB n'est pas le contrôleur en charge (CIC).

ist – Définir le bit d'état individuel

syntaxe **ist** 0 (bit d'état individuel réinitialisé)

ist 1 (bit d'état individuel activé)

ist établit le sens du bit d'état individuel (**ist**).

Vous utilisez **ist** quand le contrôleur GPIB n'est pas le contrôleur en charge (CIC) mais qu'il participe à une interrogation parallèle (parallel poll) dirigée par un appareil qui est le contrôleur en charge. Le contrôleur en charge (CIC) dirige une interrogation parallèle (parallel poll) en excitant les signaux EOI et ATN qui envoient le message "Identifier" (IDY). Bien que ce message soit actif, chaque appareil que vous avez configuré pour participer à l'interrogation répond en excitant une ligne de données GPIB prédéterminée TRUE ou FALSE, selon la valeur de son bit **ist local**. Par exemple, vous pouvez demander au contrôleur GPIB qu'il place la ligne de données DIO3 sur TRUE si **ist** est 1 et sur FALSE si **ist** est 0. De la même façon, vous pouvez lui demander de placer DIO3 sur TRUE si **ist** est 0 et sur FALSE si **ist** est 1.

Le message d'activation d'interrogation parallèle (PPE) en vigueur pour chaque appareil détermine la relation entre la valeur d'**ist**, la ligne dirigée, et le sens dans lequel la ligne est dirigée. Le contrôleur peut recevoir ce message localement par le biais de **ppc** ou à distance par le biais d'une commande du contrôleur en charge (CIC). Une fois que le message PPE s'exécute, **ist** change le sens vers lequel le contrôleur GPIB dirige la ligne lors de l'interrogation parallèle et le contrôleur GPIB peut transmettre un message d'un bit qui dépend de l'appareil au contrôleur.

llo – Verrouillage local

syntaxe **llo**

llo place tous les appareils en état de verrouillage local. Cette action réduit normalement la reconnaissance d'entrées de la face-avant de l'appareil.

llo transmet la commande de verrouillage local (LLO).

loc – Mettre le contrôleur en état local

syntaxe **loc**

loc place le contrôleur GPIB en état local en envoyant le message "Revenir à local" (RTL) s'il n'est pas verrouillé en mode à distance (indiqué par le bit d'état LOK). Utilisez **loc** pour simuler un commutateur RTL de face-avant quand vous utilisez un ordinateur pour simuler un instrument.

off – Mettre le contrôleur hors ligne*syntaxe* **off**

off place le contrôleur hors ligne. Ceci n'est nécessaire que lorsque le contrôleur est partagé avec une autre application qui utilise la bibliothèque NI-488.

ppc – Configurer l'interrogation parallèle (activer et désactiver)*syntaxe* **ppc** *octet*

ppc configure le contrôleur GPIB pour participer à une interrogation parallèle (parallel poll) en réglant son message "Activer l'interrogation locale" (LPE) à la valeur d'*octet*. Si la valeur d'*octet* est 0, le contrôleur GPIB est déconfiguré.

Chacun des 16 messages "Activer l'interrogation parallèle" (PPE) sélectionne une ligne de données GPIB (DIO1 à DIO8) et un sens (1 ou 0) que l'appareil doit utiliser quand il répond au message "Identifier" (IDY [Identify]) lors d'une interrogation parallèle (parallel poll). L'appareil interprète le message affecté et la valeur courante du bit d'état individuel (**ist**) pour déterminer si la ligne sélectionnée est placée sur TRUE ou sur FALSE. Par exemple, si PPE=0x64, la ligne DIO5 est placée sur TRUE si **ist** est 0 et sur FALSE si **ist** est 1. Si PPE=0x68, le message DIO1 PPE est en vigueur. Vous devez savoir quels messages PPE et PPD sont envoyés et déterminer ce que les réponses indiquent.

ppu – Déconfigurer l'interrogation parallèle (parallel poll)*syntaxe* **ppu**

ppu désactive tous les appareils de réponse aux interrogations parallèles.

ppu transmet la commande "Déconfigure l'interrogation parallèle" (PPU).

rpp – Effectuer une interrogation parallèle (parallel poll)*syntaxe* **rpp**

rpp effectue une interrogation parallèle d'appareils précédemment configurés en excitant les signaux ATN et EOI qui transmettent le message IDY.

rpp place la réponse à l'interrogation parallèle dans la chaîne de caractères de sortie sous forme de caractères ASCII.

rsc – Demander ou affranchir le contrôle du système

syntaxe **rsc** 0 (affranchir le contrôle du système)

rsc 1 (demander le contrôle du système)

rsc affranchit ou demande la capacité du contrôleur GPIB d'envoyer les messages "Réinitialisation de l'interface" (IFC) et "Activation à distance" (REN) aux appareils GPIB en utilisant les fonctions **sic** et **sre**. Pour que le contrôleur GPIB réponde au IFC envoyé par un autre contrôleur, le contrôleur GPIB ne doit pas être le contrôleur du système.

Dans la plupart des applications, le contrôleur GPIB est toujours le contrôleur du système. N'utilisez **rsc** que si l'ordinateur n'est pas le contrôleur du système pour la durée de l'exécution du programme.

rsv – Demander un service et/ou activer l'octet d'état d'interrogation série (serial poll)

syntaxe **rsv** *octet*

rsv établit l'octet d'état de l'interrogation série (serial poll) du contrôleur GPIB sur *octet*. Si le bit 0x40 est réglé sur *octet*, le contrôleur GPIB demande également un service de la part du contrôleur en excitant la ligne GPIB RSQ. Par exemple, si vous souhaitez exciter la ligne GPIB RSQ, envoyez le caractère ASCII @, où le bit 0x40 est établi.

Utilisez **rsv** pour demander un service de la part du contrôleur en utilisant le signal "Demander un service" (SRQ) et pour fournir un octet d'état qui dépend du système quand le contrôleur effectue une interrogation série du port GPIB.

sic – Envoyer une réinitialisation d'interface

syntaxe **sic**

sic fait que le contrôleur excite le signal IFC pendant au moins 100 ms si le contrôleur détient l'autorité de contrôleur du système. Cette action initialise le GPIB et fait que le port contrôleur constitue le contrôleur en charge (CIC). En général, utilisez **sic** quand vous souhaitez qu'un appareil devienne le CIC ou pour réinitialiser une condition d'erreur du bus.

Le signal IFC ne réinitialise que les fonctions GPIB des appareils du bus ; il ne rétablit pas les fonctions d'appareils internes. Les commandes "Réinitialiser l'appareil" (DCL) et "Réinitialiser l'appareil sélectionné" (SDC) réinitialisent les fonctions de l'appareil. Consultez la documentation de l'instrument pour déterminer l'effet de ces messages.

sre – Etablir ou réinitialiser l'activation à distance

syntaxe **sre** 0 (réinitialiser l'activation à distance)

sre 1 (établir l'activation à distance)

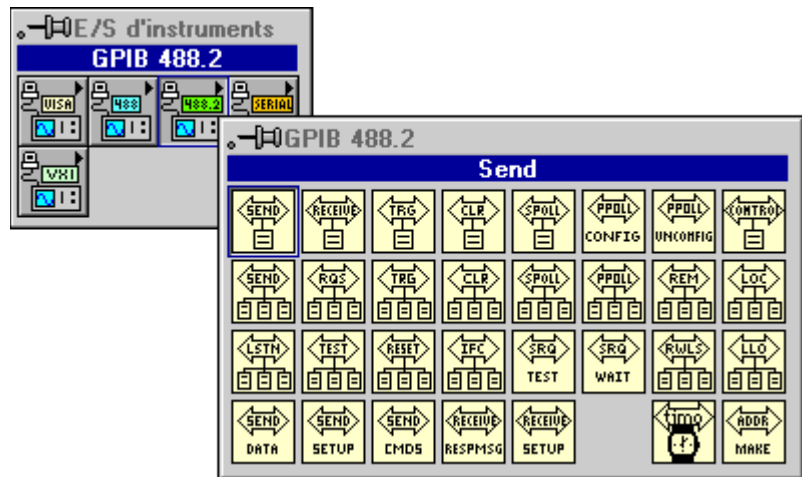
sre sert à établir ou réinitialiser la ligne GPIB REN. Les appareils surveillent la ligne REN pendant la sélection des modes de fonctionnement local et à distance. En fait, un appareil ne passe pas en mode à distance tant qu'il n'a pas reçu son adresse "Listener".

L'erreur ESAC survient si le contrôleur n'est pas le contrôleur du système.

Fonctions GPIB 488.2

Ce chapitre décrit les fonctions IEEE 488.2 (GPIB).

La figure ci-après montre la palette **GPIB 488.2** que vous pouvez ouvrir en sélectionnant **Fonctions»E/S d'instruments»GPIB 488.2**.



Pour des exemples sur l'utilisation des fonctions GPIB 488.2, reportez-vous à `exemples\instr\smp1GPIB.llb`.

Paramètres courants des fonctions GPIB 488.2

La plupart des fonctions GPIB 488.2 utilisent les paramètres suivants :

- **adresse** comprend l'adresse primaire de l'appareil GPIB avec lequel la fonction communique. Si une adresse secondaire est requise, utilisez la fonction `MakeAddr` pour établir le format correct des adresses primaire et secondaire. Sauf indication contraire, **adresse et liste d'adresses** sont respectivement de type de données entiers et tableau d'entiers.
- L'adresse primaire par défaut de la carte GPIB est 0, sans adresse secondaire. Elle est désignée comme contrôleur du système. La valeur de timeout par défaut pour les fonctions est 10 secondes. Si vous

souhaitez changer n'importe lequel de ces paramètres, employez l'utilitaire de configuration compris avec votre carte GPIB. Vous pouvez également utiliser les fonctions GPIB Init et SetTimeout pour régler l'adresse primaire et changer la valeur de timeout par défaut en cours d'exécution. Cependant ces fonctions n'affectent que l'interface quand vous les utilisez avec LabVIEW. Pour plus d'informations, reportez-vous à la documentation livrée avec votre interface matérielle.

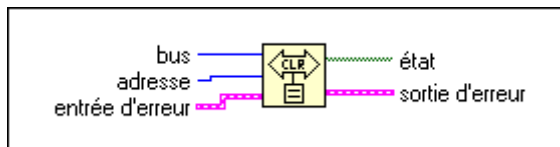
- **bus** se rapporte au numéro du bus GPIB. Si vous n'avez qu'une interface GPIB dans votre ordinateur, le numéro de bus par défaut est 0. Pour des interfaces GPIB supplémentaires, reportez-vous aux instructions sur l'installation logicielle jointes à votre carte GPIB.
- **nombre d'octets** se rapporte au nombre d'octets transmis par le GPIB.
- **état** est un tableau booléen où chaque bit décrit un état du contrôleur GPIB. Si une erreur survient, les fonctions du GPIB activent le bit 15. **Erreur GPIB** n'est valide que si le bit 15 d'**état** est activé.
- **entrée d'erreur; sortie d'erreur.** Reportez-vous à la section *Paramètres des fonctions traditionnelles du GPIB* au chapitre 34, *Fonctions GPIB traditionnelles*.

Descriptions des fonctions GPIB 488.2 (Fonctions d'appareil unique)

Les fonctions d'appareil unique effectuent les E/S GPIB et les opérations de contrôle avec un seul appareil GPIB. En général, chaque fonction accepte une adresse d'appareil unique comme une de ses entrées.

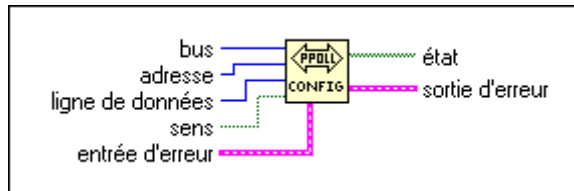
DevClear

Réinitialise un appareil unique. Pour envoyer le message "Réinitialiser l'appareil sélectionné" (SDC) à plusieurs appareils, utilisez la fonction DevClearList.



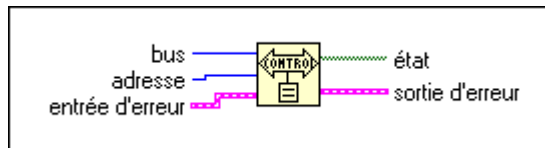
PPollConfig

Configure un appareil pour des interrogations parallèles (parallel polls).



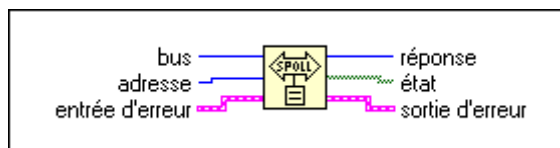
PassControl

Passes le contrôle à un autre appareil muni de la capacité de Contrôleur.



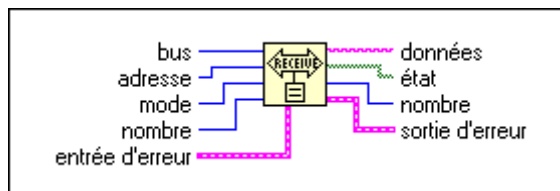
ReadStatus

Effectue une interrogation en série (serial poll) d'un appareil unique pour obtenir son octet d'état.



Receive

Lit des octets de données d'un appareil GPIB.



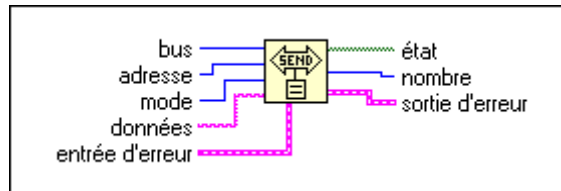
La fonction Receive se termine quand la fonction exécute l'une des actions suivantes :

- lit le nombre d'octets requis
- détecte une erreur

- dépasse la limite temporelle
- détecte le message END (EOI activé)
- détecte le caractère EOS (en supposant que la valeur de **mode** active cette option)

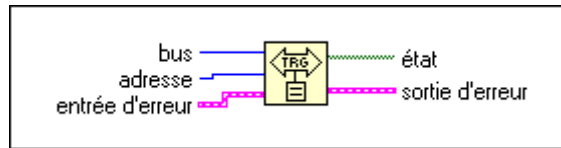
Send

Envoie des octets de données à un appareil GPIB unique.



Trigger

Déclenche un appareil unique. Pour envoyer un seul message qui déclenche plusieurs appareils GPIB, utilisez la fonction TriggerList.

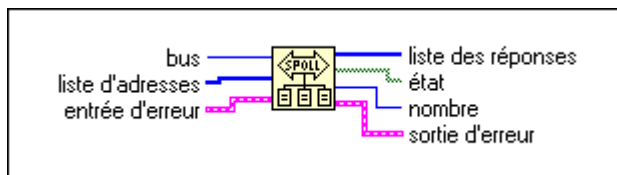


Descriptions des fonctions d'appareils multiples GPIB 488.2

Les fonctions d'appareils multiples effectuent les E/S GPIB et les opérations de contrôle avec plusieurs appareils GPIB en même temps. En général, chaque fonction accepte un tableau d'adresses comme une de ses entrées.

AIISPoll

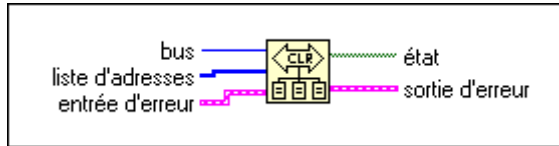
Effectue une interrogation en série (serial poll) de tous les appareils.



Bien que la fonction AllSPoll soit suffisamment générale pour effectuer une interrogation en série de plusieurs appareils GPIB, utilisez la fonction ReadStatus quand vous souhaitez effectuer une interrogation en série d'un seul appareil GPIB.

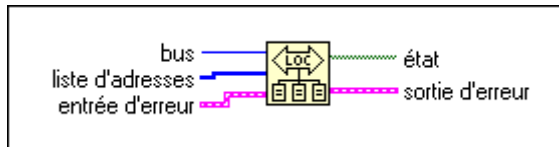
DevClearList

Réinitialise plusieurs appareils simultanément.



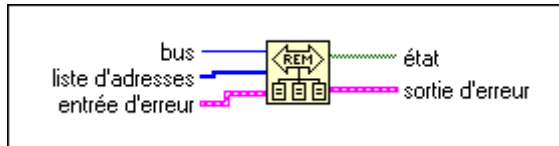
EnableLocal

Active le mode local pour plusieurs appareils.



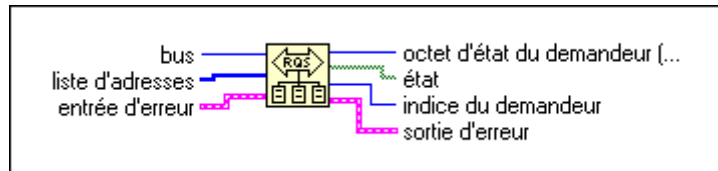
EnableRemote

Active la programmation à distance de plusieurs appareils GPIB.



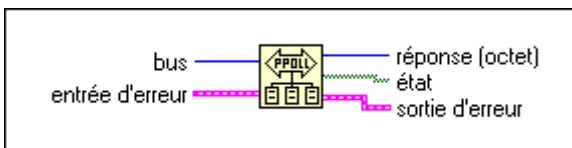
FindRQS

Détermine quel appareil demande un service.



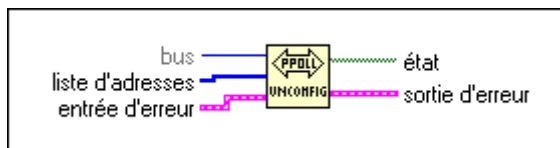
PPoll

Effectue une interrogation en série.



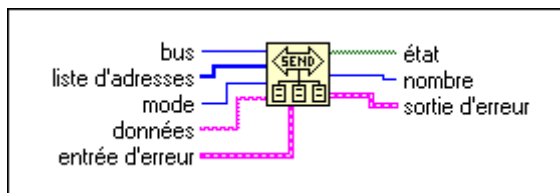
PPollUnconfig

Déconfigure des appareils pour des interrogations parallèles. La fonction déconfigure les appareils GPIB dont les adresses sont comprises dans le tableau de liste d'adresses pour des interrogations parallèles (parallel polls) ; c'est-à-dire qu'ils ne participent plus aux interrogations.



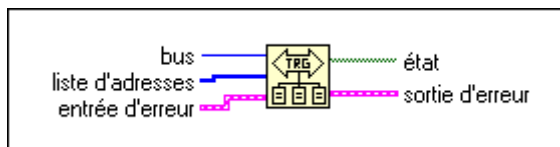
SendList

Envoie des octets de données à plusieurs appareils GPIB. Cette fonction est comparable à Send (envoyer), excepté que SendList envoie des données à plusieurs Auditeurs (Listeners) en une seule transmission.



TriggerList

Déclenche plusieurs appareils simultanément.

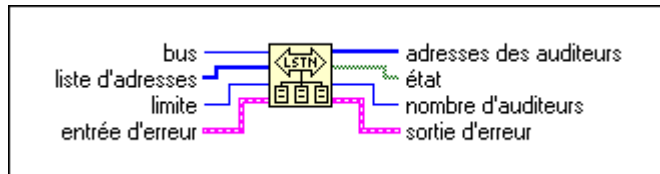


Descriptions des fonctions de gestion du bus GPIB 488.2

Les fonctions de gestion du bus effectuent des fonctions à l'échelle du système ou indiquent l'état à l'échelle du système.

FindLstn

Trouve tous les Auditeurs (Listeners) sur le GPIB. Normalement, vous utilisez cette fonction pour détecter la présence d'appareils à des adresses précises car la plupart des appareils GPIB peuvent écouter. D'habitude, quand vous les détectez, vous pouvez interroger les appareils pour déterminer leur identité.

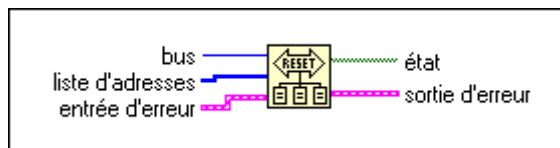


ResetSys

Effectue l'initialisation du bus, l'initialisation de l'échange de message et l'initialisation de l'appareil. D'abord, la fonction active la ligne "Activation à distance" (REN), suivi de "Réinitialiser l'interface" (IFC), désadressant ainsi tous les appareils et configurant la carte GPIB (le contrôleur du système) comme Contrôleur en charge (CIC).

Ensuite, la fonction envoie le message "Réinitialiser l'appareil" (DCL) à tous les appareils connectés. Ceci garantit que les appareils compatibles IEEE 488.2 peuvent recevoir le message "Réinitialiser" (RST) qui suit.

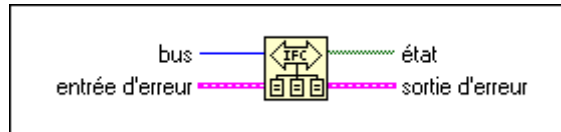
Finalement, la fonction envoie le message *RST à tous les appareils dont les adresses figurent dans le tableau de la liste d'adresses. Ce message initialise les fonctions spécifiques à l'appareil au sein de chaque appareil.



SendIFC

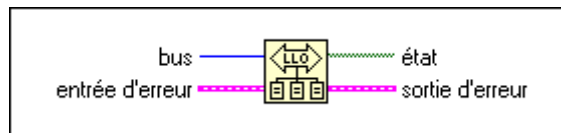
Réinitialise les fonctions GPIB avec “Réinitialiser l’interface” (IFC). Quand vous émettez le message IFC, les fonctions d’interface de tous les appareils connectés reviennent à leurs états réinitialisés.

Utilisez cette fonction pour effectuer l’initialisation du GPIB. Elle force la carte GPIB à agir comme Contrôleur GPIB et garantit que les appareils connectés sont tous désadressés et que les fonctions d’interface des appareils sont en état de veille.



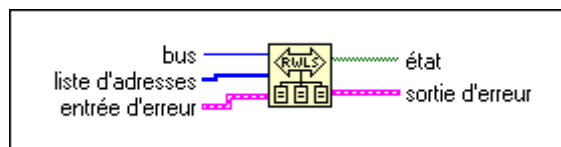
SendLLO

Envoie le message “Verrouillage local” (LLO) à tous les appareils. Quand la fonction envoie le message GPIB “Verrouillage local”, un appareil ne peut pas sélectionner indépendamment l’état local ou à distance. Tant que “Verrouillage local” est en vigueur, seul le contrôleur peut modifier l’état local ou à distance des appareils en envoyant les messages GPIB appropriés. N’utilisez SendLLO que dans des situations locales/à distance exceptionnelles, en particulier quand vous devez verrouiller tous les appareils en état de programmation locale. Utilisez la fonction SetRWLS quand vous souhaitez placer des appareils en “Mode à distance avec état de verrouillage” (Remote Mode with Lockout State).



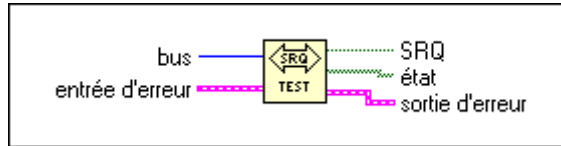
SetRWLS

Place des appareils spécifiques en “Mode à distance avec état de verrouillage” (Remote Mode with Lockout State). La fonction envoie “Activation à distance” (REN) aux appareils GPIB listés dans la **liste d’adresses**. Elle place également tous les appareils en Etat de verrouillage, ce qui évite qu’ils retournent d’eux-mêmes au mode de programmation locale sans l’intervention du Contrôleur.



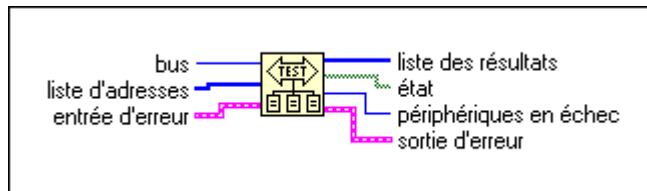
TestSRQ

Détermine l'état actuel de la ligne SRQ. Le format de cette fonction ressemble à celui de la fonction WaitSRQ, excepté que WaitSRQ se met en suspens en attendant une occurrence de SRQ et que TestSRQ retourne immédiatement l'état SRQ actuel.



TestSys

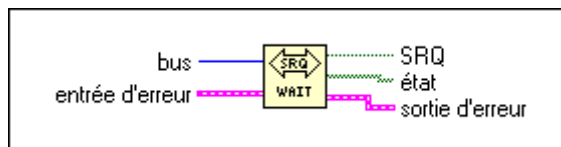
Commande à plusieurs appareils d'effectuer des auto-tests IEEE 488.2.



WaitSRQ

Attend qu'un appareil active une "Demande de service" (Service Request). La fonction suspend l'exécution jusqu'à ce qu'un appareil connecté au GPIB active la ligne de demande de service (SRQ).

Le format de cette fonction est comparable à celui de TestSRQ, excepté que TestSRQ retourne immédiatement l'état SRQ, tandis que WaitSRQ suspend le programme pendant la durée du timeout (mais pas plus) en attendant qu'une SRQ survienne.

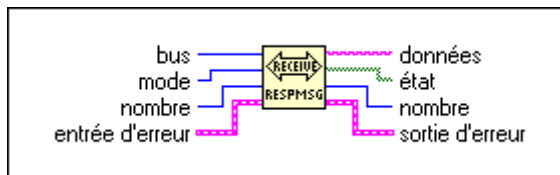


Descriptions des fonctions d'E/S de bas niveau GPIB 488.2

Les fonctions de bas niveau vous permettent de créer un programme plus spécifique et détaillé que les fonctions de haut niveau. Utilisez les fonctions de bas niveau lors de situations exceptionnelles ou quand elles requièrent plus de flexibilité.

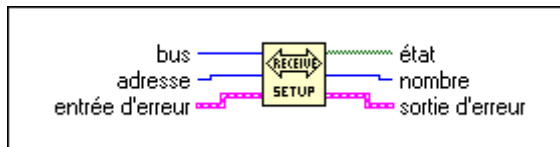
RcvRespMsg

Lit des octets de données d'un appareil précédemment adressé. Cette fonction suppose qu'une autre fonction, comme ReceiveSetup, Receive ou SendCmds, a déjà adressé les Transmetteurs (Talkers) et les Auditeurs (Listeners) GPIB. Utilisez RcvRespMsg spécifiquement pour sauter l'étape d'adressage dans la gestion du GPIB. Normalement, vous devez utiliser la fonction Receive pour effectuer toute la séquence d'adressage et pour recevoir ensuite des octets de données.



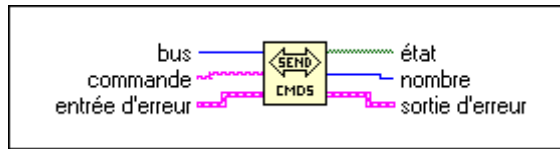
ReceiveSetup

Prépare un appareil pour envoyer des octets de données et prépare la carte GPIB pour lire des octets de données. Après avoir appelé cette fonction, vous pouvez utiliser une fonction telle que RcvRespMsg pour transférer les données du Transmetteur (Talker). De cette façon, vous éliminez le besoin de réadresser les appareils entre blocs de lecture.



SendCmds

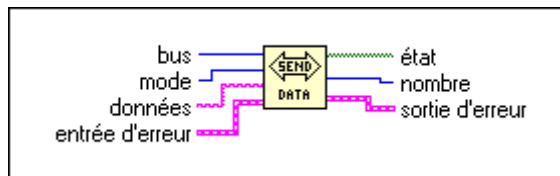
Envoie des octets de commande GPIB.



Il n'est pas nécessaire, en général, d'utiliser SendCmds pour une opération GPIB. Utilisez cette fonction quand vous devez envoyer des commandes spécialisées sur le GPIB.

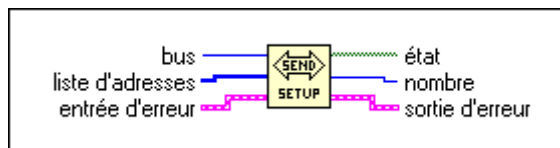
SendDataBytes

Envoie des octets de données à des appareils précédemment adressés.



SendSetup

Prépare des appareils spécifiques à recevoir des octets de données. En général, vous devez faire suivre un appel à cette fonction par un appel à une fonction comme SendDataBytes pour transférer effectivement des données aux auditeurs. Cette séquence élimine la nécessité de réadresser les appareils entre blocs d'envois.

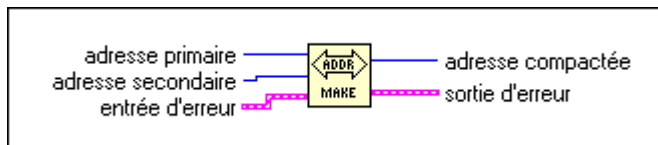


Descriptions des fonctions générales GPIB 488.2

Les fonctions générales sont utiles lors de situations particulières.

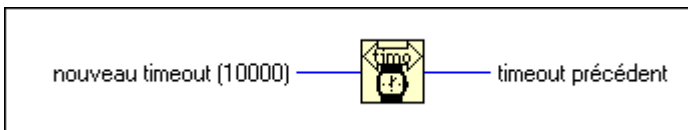
MakeAddr

Combine l'**adresse primaire** et l'**adresse secondaire** dans une **adresse compactée** spécialement formatée pour des appareils qui exigent une adresse primaire et une adresse secondaire GPIB.



SetTimeout

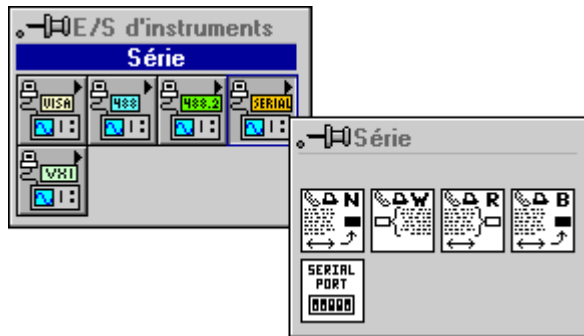
Change la durée du timeout global pour toutes les fonctions GPIB 488.2. Cette fonction règle également la durée du timeout par défaut pour toutes les fonctions GPIB.



Vis de Port série

Ce chapitre décrit les VIs pour les opérations de port série.

La figure suivante montre la palette **Série** que vous pouvez ouvrir en sélectionnant **Fonctions»E/S d'instruments»Série**.



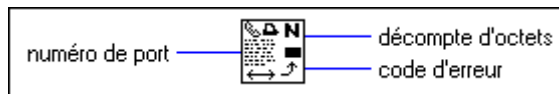
Pour des exemples de l'emploi des VIs de port série, reportez-vous à `examples\instr\smp1ser1.llb`.

Descriptions des VIs Port série

Les VIs Port série suivants sont disponibles.

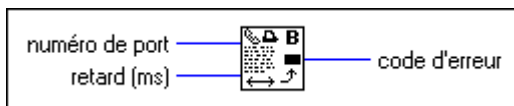
Bytes at Serial Port

Retourne le nombre d'octets spécifié du buffer d'entrée du port série indiqué par **numéro de port**.



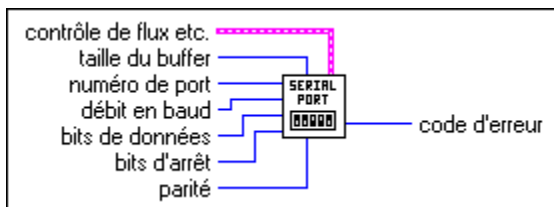
Serial Port Break

Envoie un arrêt sur le port de sortie spécifié par **numéro de port** pour une durée au moins aussi longue que l'exige l'entrée **retard**.



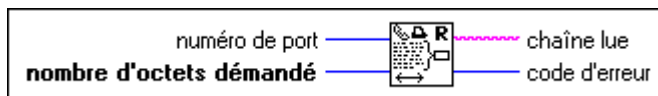
Serial Port Init

Initialise le port série sélectionné selon les paramètres spécifiés.



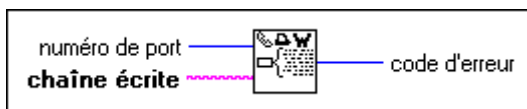
Serial Port Read

Lit le nombre de caractères spécifié par **nombre d'octets demandé** sur le port série indiqué par **numéro de port**.



Serial Port Write

Ecrit des données spécifiées par **chaîne à écrire** sur le port série indiqué par **numéro de port**.



Partie IV

VIs d'analyse

La Partie IV, *VIs d'analyse*, décrit les VIs d'analyse. Cette partie contient les chapitres suivants :

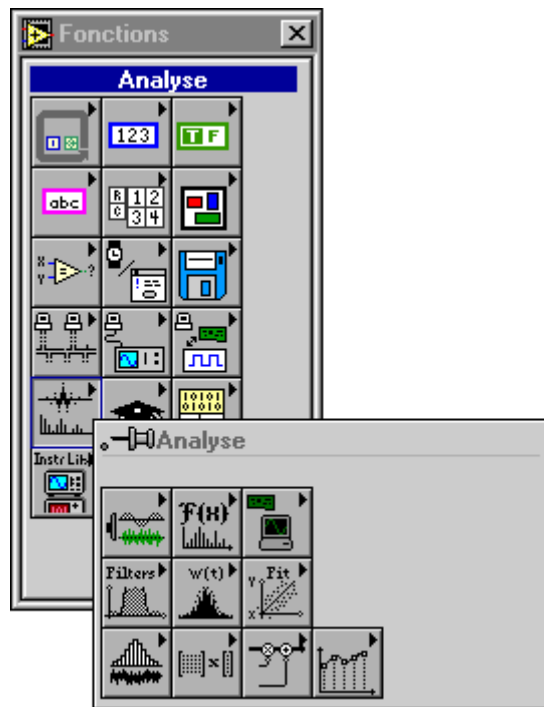
- Le chapitre 37, *Introduction à l'analyse dans LabVIEW*, introduit les VIs d'analyse de LabVIEW. Il présente également une description de l'organisation des VIs, les instructions pour accéder aux VIs, l'obtention d'aide en ligne et une description du rapport d'erreur des VIs d'analyse.
- Le chapitre 38, *VIs de génération de signaux*, décrit les VIs qui génèrent les tableaux unidimensionnels avec des motifs de formes d'onde spécifiques.
- Le chapitre 39, *VIs de traitement des signaux numériques*, décrit les VIs qui traitent et analysent des signaux acquis ou simulés. Les VIs de traitement des signaux numériques effectuent des transformations et des analyses du domaine fréquentiel, des analyses du domaine temporel et autres transformées telles que les transformées de Fourier, Hartley et Hilbert.
- Le chapitre 40, *VIs de mesure*, décrit les VIs de mesure, qui sont structurés pour effectuer des analyses basées sur les DFT et FFT sur des signaux acquis par des applications de mesure de fréquence comme celles effectuées par des instruments de mesure de fréquence typiques tels que les analyseurs de signaux dynamiques.
- Le chapitre 41, *VIs Filtres*, décrit les VIs utilisés pour implémenter des filtres IIR, FIR et non linéaires.
- Le chapitre 42, *VIs Fenêtres*, décrit les VIs utilisés pour implémenter des fenêtres de lissage.
- Le chapitre 43, *VIs d'ajustement de courbe*, décrit les VIs utilisés pour effectuer des ajustements de courbes et des analyses de régression.
- Le chapitre 44, *VIs Probabilités et statistiques*, décrit les VIs utilisés pour exécuter des fonctions de probabilité, de statistiques descriptives, d'analyse de variance et d'interpolation.

- Le chapitre 45, *Vis d'algèbre linéaire*, décrit les VIs utilisés pour effectuer des calculs et des analyses sur des matrices réelles et complexes.
- Le chapitre 46, *Vis d'opérations sur tableaux*, décrit les VIs utilisés pour effectuer des opérations sur tableaux numériques classiques à une ou deux dimensions.
- Le chapitre 47, *Vis de méthodes numériques supplémentaires*, décrit les VIs qui implémentent des méthodes numériques pour effectuer des recherches de racine, des intégrations numériques et des détections de crêtes.

Introduction à l'analyse dans LabVIEW

Ce chapitre présente les VIs d'analyse de LabVIEW, une description de l'organisation de ces VIs, des instructions sur l'accès aux VIs et sur l'obtention d'aide en ligne, et une description des erreurs des VIs d'analyse.

Pour ouvrir la palette **Analyse** à partir du diagramme, choisissez **Fonctions»Analyse** et naviguez dans les menus hiérarchiques pour sélectionner le VI souhaité. Vous pouvez placer l'icône du VI dans le diagramme et le câbler ensuite.



Systeme de developpement complet

La bibliothèque des VIs d'analyse de base est un sous-ensemble de la bibliothèque de VIs d'analyse avancée disponible dans le Système de développement complet. La bibliothèque d'analyse de base comprend des VIs d'analyse statistique, d'algèbre linéaire et d'analyse numérique. La bibliothèque d'analyse avancée comprend d'autres VIs de ce type ainsi que des VIs de génération de signaux, des algorithmes de domaine temporel et fréquentiel, des routines de fenêtrage, des filtres numériques, des évaluations et des régressions.

Si les VIs de la bibliothèque d'analyse de base ne satisfont pas à vos besoins, vous pouvez ajouter les bibliothèques d'analyse avancée de LabVIEW à l'ensemble de base du G. Après la mise à jour, vous aurez tous les outils d'analyse disponibles dans le Système de développement complet.

Reportez-vous aux chapitre 38 à 47, qui présentent chaque sous-palette d'analyse, pour obtenir plus d'informations sur l'accès à une fonction particulière ou à une palette de VIs.

Vue d'ensemble des VIs d'analyse

Les VIs d'analyse de LabVIEW traitent avec efficacité des blocs d'informations représentés sous forme numérique. Ils traitent les éléments importants suivants :

- Génération de patterns
- Traitement des signaux numériques
- Analyse de mesures
- Filtrage numérique
- Fenêtres de lissage
- Probabilités et analyse statistique
- Ajustement de courbe
- Algèbre linéaire
- Analyse numérique

Les VIs d'analyse effectuent des opérations numériques en utilisant l'unité centrale de traitement (CPU) et un coprocesseur à virgule flottante (FPU). De nombreux VIs tirent parti des capacités de traitement concurrent du CPU et du FPU, minimisant ainsi le temps d'exécution des tâches d'analyse de données.

Les VIs d'analyse utilisent des algorithmes de traitement de données intégré. Autrement dit, les algorithmes affectent un espace de données minimal et traitent les données au sein de cet espace. Le traitement intégré minimise les besoins en mémoire, vous permettant de traiter des blocs de données plus larges. La seule limite de mémoire de ces VIs est la quantité de RAM disponible dans votre ordinateur. Reportez-vous au *Manuel de l'utilisateur LabVIEW* pour des instructions sur la configuration de l'allocation de mémoire dans LabVIEW.

Les VIs d'analyse sont suffisamment puissants pour que les utilisateurs expérimentés puissent rapidement et efficacement construire des applications avancées d'analyse. En même temps, ils sont suffisamment simples pour les débutants souhaitant analyser des données sans nécessairement être experts en DSP, filtres numériques, statistiques ou analyse numérique.

Organisation des VIs d'analyse

Après l'installation, les dix bibliothèques de VIs d'analyse apparaissent dans la palette **Fonctions**. La palette **Analyse** comporte les sous-palettes suivantes :

- **Génération de signaux** comprend des VIs utilisés pour générer des patterns numériques et des formes d'ondes.
- **Traitement des signaux numériques** comprend des VIs utilisés pour effectuer des transformations et des analyses du domaine fréquentiel et des analyses du domaine temporel, et d'autres transformées telles que celles de Hartley et Hilbert.
- **Mesure** comprend des VIs utilisés pour effectuer des opérations sur des mesures comme les spectres unilatéraux, le fenêtrage mis à l'échelle et les calculs de fréquence et de puissance de crête.
- **Filtres** comprend des VIs utilisés pour effectuer des fonctions de filtrage numérique IIR, FIR et non linéaires.
- **Fenêtres** comprend des VIs utilisés pour effectuer le fenêtrage de données.
- **Probabilités et statistiques** comprend des VIs utilisés pour effectuer des fonctions statistiques descriptives, comme identifier l'écart-type ou moyen d'un ensemble de données, ainsi que des fonctions de statistiques déductives de probabilité et l'analyse de variance (ANOVA).
- **Ajustement de courbe** comprend des VIs utilisés pour effectuer des fonctions d'ajustement de courbe et des interpolations.

- **Algèbre linéaire** comprend des VIs utilisés pour effectuer des fonctions algébriques sur des matrices et des vecteurs réels et complexes.
- **Opérations sur les tableaux** comprend des VIs utilisés pour effectuer des opérations communes de tableau numérique, unidimensionnel et bidimensionnel, comme la mise à l'échelle et l'évaluation linéaire.
- **Méthodes numériques supplémentaires** comprend des VIs qui utilisent des méthodes numériques pour effectuer des opérations de recherche, d'intégration numérique et de détection de pics.

Vous pouvez réorganiser les dossiers et les VIs selon vos préférences et vos applications. Vous pouvez également reconstruire la structure originale en enlevant des VIs de votre unité de disque dur et en les réinstallant ensuite à partir des disquettes de distribution.

Nomenclature utilisée dans ce manuel

Pour vous aider à identifier le type de paramètres et d'opérations, cette section du manuel utilise la nomenclature suivante, sauf indication contraire dans une description de VI. Bien qu'il existe quelques opérations et fonctions scalaires, la plupart des VIs d'analyse traitent de grands blocs de données sous forme de tableaux 1D (ou vecteurs) et 2D (ou matrices).

Les minuscules représentent des scalaires ou des constantes. Par exemple,

$$\begin{aligned} a, \\ \pi, \\ b = 1,234 \end{aligned}$$

Les majuscules représentent des tableaux. Par exemple,

$$\begin{aligned} X, \\ A, \\ Y = a X + b \end{aligned}$$

En général, X et Y indiquent des tableaux 1D, tandis que A , B et C représentent des matrices.

Les indices de tableaux de LabVIEW sont en base zéro. L'indice du premier élément dans le tableau, quelle que soit sa dimension, est zéro. La séquence de chiffres suivante représente un tableau 1D X comportant n éléments.

$$X = \{x_0, x_1, x_2, \dots, x_{n-1}\}$$

La quantité scalaire suivante représente le $i^{\text{ème}}$ élément de la séquence X .

$$x_i, \quad 0 \leq i < n$$

Le premier élément dans la séquence est x_0 et le dernier élément dans la séquence est x_{n-1} , pour un total de n éléments.

La séquence de chiffres suivante représente un tableau 2D comportant n rangées et m colonnes.

$$A = \begin{bmatrix} a_{00} & a_{01} & a_{02} & \dots & a_{0m-1} \\ a_{10} & a_{11} & a_{12} & \dots & a_{1m-1} \\ a_{20} & a_{21} & a_{22} & \dots & a_{2m-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n-10} & a_{n-11} & a_{n-12} & \dots & a_{n-1m-1} \end{bmatrix}$$

Le nombre total d'éléments dans le tableau 2D est le produit de n et m . Le premier indice correspond au numéro de rangée et le second indice correspond au numéro de colonne. La quantité scalaire suivante représente l'élément situé dans la $i^{\text{ème}}$ rangée et la $j^{\text{ème}}$ colonne.

$$a_{ij}, 0 \leq i < n \text{ et } 0 \leq j < m$$

Le premier élément dans A est a_{00} et le dernier élément est a_{n-1m-1} .

Sauf indication contraire, ce manuel utilise les notations simplifiées suivantes pour les opérations de tableau.

La définition des éléments d'un tableau en une constante scalaire est représentée par

$$X = a$$

qui correspond à la séquence

$$X = \{a, a, a, \dots, a\}$$

et est utilisée au lieu de

$$x_i = a, \text{ pour } i = 0, 1, 2, \dots, n-1$$

La multiplication des éléments d'un tableau par une constante scalaire est représentée par

$$Y = aX$$

qui correspond à la séquence

$$Y = \{ax_0, ax_1, ax_2, \dots, ax_{n-1}\}$$

et est utilisée au lieu de

$$y_i = ax_i, \text{ pour } i = 0, 1, 2, \dots, n-1$$

De même, la multiplication d'un tableau 2D par une constante scalaire est représentée par

$$B = kA$$

qui correspond à la séquence

$$B = \begin{bmatrix} ka_{00} & ka_{01} & ka_{02} & \dots & ka_{0m-1} \\ ka_{10} & ka_{11} & ka_{12} & \dots & ka_{1m-1} \\ ka_{20} & ka_{21} & ka_{22} & \dots & ka_{2m-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ ka_{n-10} & ka_{n-11} & ka_{n-12} & \dots & ka_{n-1m-1} \end{bmatrix}$$

et est utilisée au lieu de

$$b_{ij} = ka_{ij}, \text{ pour } i = 0, 1, 2, \dots, n-1 \text{ et } j = 0, 1, 2, \dots, m-1$$

Les tableaux vides sont possibles dans LabVIEW. Un tableau sans élément est un tableau vide représenté par

$$\text{Vide} = \text{NULL} = \emptyset = \{ \}$$

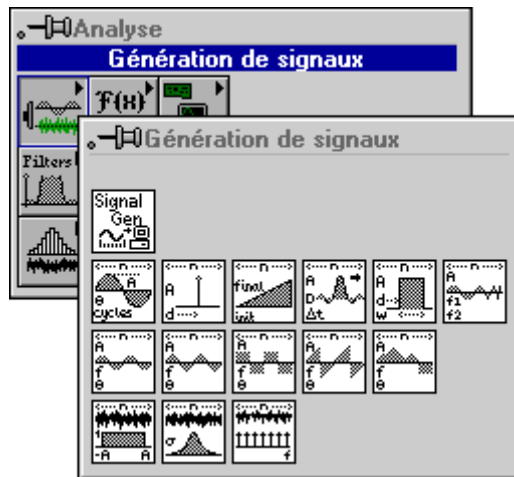
En général, les opérations sur des tableaux vides donnent des tableaux de sortie vides ou des résultats indéterminés.

Vis de génération de signaux

Ce chapitre décrit les VIs utilisés pour générer des tableaux à une dimension avec des motifs de formes d'ondes spécifiques.

Vous pouvez combiner ces VIs avec les fonctions arithmétiques figurant au chapitre 4, *Fonctions numériques*, pour générer des formes d'ondes plus complexes. Par exemple, si vous souhaitez générer une impulsion modulée en amplitude, vous pouvez multiplier un motif d'impulsion par un motif sinusoïdal.

Pour ouvrir la palette **Génération de signaux**, sélectionnez **Fonctions»Analyse»Génération de signaux**. L'illustration ci-après montre les options disponibles dans la palette **Génération de signaux**.



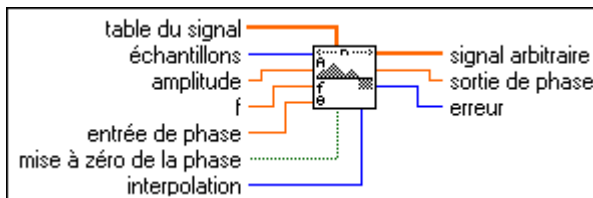
Pour des exemples sur l'emploi des VIs de génération de signaux, reportez-vous aux exemples dans `exemples\analysis\sigxmpl.llb`.

Descriptions des VIs de génération de signaux

Les VIs de génération de signaux suivants sont disponibles.

Signal arbitraire (Arbitrary Wave.vi)

Génère un tableau constituant un signal arbitraire.



Si la séquence y représente le **signal arbitraire**, le VI génère alors un motif selon la formule suivante :

$$y[i] = a * \text{arb}(\text{phase}[i]), \text{ pour } i = 0, 1, 2, \dots, n-1$$

où a représente l'**amplitude**, n représente le nombre d'**échantillons**,

$$\text{arb}(\text{phase}[i]) = \text{TS}((\text{phase}[i] \bmod 360) * m / 360)$$

où m représente la taille du **tableau du signal**.

Si **interpolation** = 0 (pas d'interpolation), alors $\text{TS}(x) = \text{tableau du signal}[\text{int}(x)]$.

Si **interpolation** = 1 (interpolation linéaire), alors $\text{TS}(x)$ est égal à la valeur interpolée linéairement de **tableau du signal** $[\text{int}(x)]$ et **tableau du signal** $[(\text{int}(x)+1) \bmod m]$.

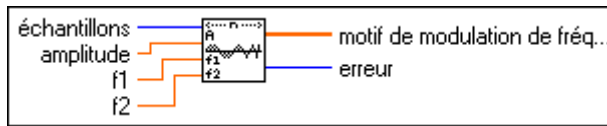
$\text{phase}[i] = \text{phase_initiale} + \mathbf{f} * 360.0 * i$, où \mathbf{f} représente la fréquence en unités normalisées de cycles/échantillon, phase_initiale représente l'**entrée de phase** si **mise à zéro de la phase** est vraie, ou la **sortie de phase** de l'exécution précédente de ce VI si **mise à zéro de la phase** est fausse.

Le VI est ré-entrant et vous pouvez donc l'utiliser pour simuler une acquisition en continu d'un générateur de fonction de signal arbitraire. Si la commande d'entrée **mise à zéro de la phase** est fausse, les appels ultérieurs à une instance spécifique de ce VI produisent un tableau de sortie **Signal arbitraire** comprenant les **échantillons** suivants du signal arbitraire.

sortie de phase est réglée sur $\text{phase}[n]$, et ce VI ré-entrant utilise cette valeur comme nouvelle **entrée de phase** si **mise à zéro de la phase** est fausse la prochaine fois que le VI s'exécute.

Motif de modulation de fréquence (Chirp Pattern.vi)

Génère un tableau représentant un motif de modulation de fréquence.



Si la séquence Y représente le **Motif de modulation de fréquence**, le VI génère le motif selon la formule suivante :

$$y_i = A * \sin((a/2 i + b) i), \text{ pour } i = 0, 1, 2, \dots, n-1$$

où A représente l'**amplitude**, $a = 2\pi(f2-f1)/n$, $b = 2\pi f1$, $f1$ représente la fréquence initiale en unités normalisées de cycles/échantillon, $f2$ représente la fréquence finale en unités normalisées de cycles/échantillon et n représente le nombre d'**échantillons**.

Bruit blanc gaussien (Gaussian White Noise.vi)

Génère un motif pseudo-aléatoire de distribution gaussienne dont le profil statistique est $(\mu, \sigma) = (0, s)$, où s représente la valeur absolue de l'**écart-type** spécifié.



Pour générer le motif, le VI utilise une version modifiée de l'algorithme de génération de nombres aléatoires Cycle-très-long qui se base sur le théorème limite centrale. Etant donné que la fonction de densité de probabilité, $f(x)$, du **Motif bruit gaussien** de distribution gaussienne est :

$$f(x) = \frac{1}{\sqrt{2\pi}s} e^{-\left(\frac{1}{2}\right)\left(\frac{x}{s}\right)^2}$$

où s représente la valeur absolue de l'**écart-type** spécifié et dont vous pouvez calculer les valeurs attendues, $E\{\bullet\}$, en utilisant la formule :

$$E(x) = \int_{-\infty}^{\infty} x f(x) dx$$

alors la valeur attendue moyenne, μ , et la valeur attendue de l'écart-type, σ , de la séquence pseudo-aléatoire sont :

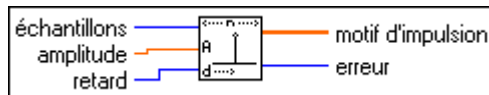
$$\mu = E\{x\} = 0$$

$$\sigma = [E\{(x - \mu)^2\}]^{1/2} = s$$

La séquence pseudo-aléatoire produit approximativement 2^{90} échantillons avant que le motif ne se répète.

Motif d'impulsion Dirac (Impulse Pattern.vi)

Génère un tableau représentant un motif d'impulsion Dirac.



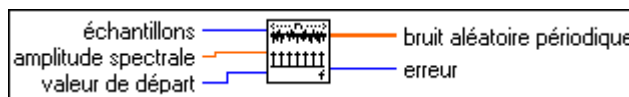
Si **Motif d'impulsion Dirac** est représenté par la séquence X, le VI génère le motif selon la formule suivante :

$$x_i = \begin{cases} a & \text{si } i = d \text{ pour } i = 0, 1, 2, \dots, n-1 \\ 0 & \text{ailleurs} \end{cases}$$

où a représente l'**amplitude**, d représente le **retard**, et n représente le nombre d'**échantillons**.

Bruit aléatoire périodique (Periodic Random Noise.vi)

Génère un tableau représentant un bruit aléatoire périodique (PRN).



Le tableau de sortie comprend toutes les fréquences qui peuvent être représentées par un nombre entier de cycles dans le nombre d'**échantillons** demandés. Chaque composante du domaine des fréquences est associée à une **amplitude spectrale** et une phase aléatoire.

Vous pouvez vous représenter le tableau de sortie de ce VI comme un cumul de signaux sinusoïdaux de mêmes amplitudes mais de phases aléatoires. L'unité d'**amplitude spectrale** est la même que la sortie **Bruit aléatoire périodique** et représente une mesure linéaire de l'amplitude, similaire aux autres VIs de génération de signaux.

Le VI génère la même séquence aléatoire périodique pour une valeur positive **source** donnée. Le VI ne réinitialise pas la génération de phase aléatoire si **source** est négative.

La séquence de sortie est limitée par une amplitude égale à

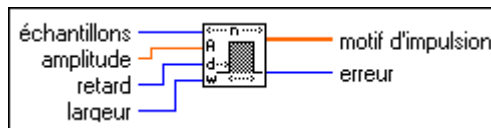
$$\text{amplitude spectrale} * \frac{\text{échantillons}}{2}$$

Vous pouvez utiliser ce VI pour calculer la réponse en fréquence d'un système linéaire lors d'un enregistrement temporel au lieu de calculer la moyenne de la réponse en fréquence de plusieurs enregistrements temporels, comme cela est nécessaire avec les sources de bruit aléatoire non périodique.

Il n'est pas nécessaire de fenêtrer le bruit aléatoire périodique avant d'effectuer l'analyse spectrale ; le bruit aléatoire périodique est muni d'un auto-fenêtrage et n'a pas de fuites spectrales car le bruit aléatoire périodique ne comprend que des sinusoïdes de cycle entier.

Motif d'impulsion (Pulse Pattern.vi)

Génère un tableau représentant un motif d'impulsion.



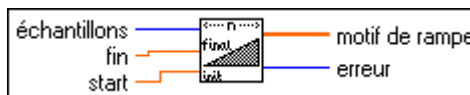
Si la séquence X représente un **Motif d'impulsion**, le VI génère le motif selon la formule suivante :

$$x_i = \begin{cases} a & \text{si } d \leq i < (d + w) \text{ pour } i = 0, 1, 2, \dots, n-1 \\ 0.0 & \text{ailleurs} \end{cases}$$

où a représente l'**amplitude**, d représente le **retard**, w représente la **largeur** et n représente le nombre d'**échantillons**.

Motif de rampe (Ramp Pattern.vi)

Génère un tableau représentant un motif de rampe.



Si la séquence X représente le **Motif de rampe**, le VI génère le motif selon la formule :

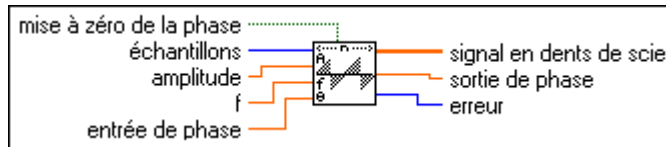
$$x_i = x_0 + i\Delta x \text{ pour } i = 0, 1, 2, \dots, n-1$$

où $\Delta x = \frac{x_{n-1} - x_0}{n - 1}$, x_{n-1} représente **fin**, x_0 représente **début** et n représente le nombre d'**échantillons**.

Le VI n'impose pas de conditions sur la relation entre **début** et **fin**. Il peut ainsi générer un motif montant ou descendant.

Signal en dents de scie (Sawtooth Wave.vi)

Génère un tableau représentant un signal en dents de scie.



Si la séquence Y représente le **signal en dents de scie**, le VI génère le motif selon la formule suivante :

$$y[i] = a * \text{dents de scie}(\text{phase}[i]), \text{ pour } i = 0, 1, 2, \dots, n-1$$

où a représente l'**amplitude**, n représente le nombre d'**échantillons**,

$$\text{dents de scie}(\text{phase}[i]) = \begin{cases} \frac{p}{180} & 0 \leq p < 180 \\ \frac{p}{180} - 2.0 & 180 \leq p < 360 \end{cases}$$

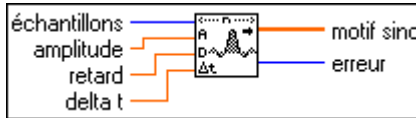
$p = \text{phase}[i] \text{ modulo } 360.0$, $\text{phase}[i] = \text{phase_initiale} + \mathbf{f} * 360.0 * i$, \mathbf{f} représente la fréquence en unités normalisées de cycles/échantillon ; phase_initiale représente l'**entrée de phase** si **mise à zéro de la phase** est vraie ; ou la **sortie de phase** de l'exécution précédente de cette instance du VI si **mise à zéro de la phase** est fausse.

Le VI est ré-entrant et vous pouvez donc l'utiliser pour simuler l'acquisition en continu d'un générateur de fonction de signal en dents de scie. Si la commande d'entrée **mise à zéro de la phase** est fausse, les appels ultérieurs à l'instance spécifique du VI produisent le tableau de sortie **signal en dents de scie** comprenant les **échantillons** suivants du signal en dents de scie.

sortie de phase est réglée sur $phase[n]$, et ce VI ré-entrant utilise cette valeur comme sa nouvelle **entrée de phase** si **mise à zéro de la phase** est fausse la prochaine fois que le VI s'exécute.

Motif sinc (Sinc Pattern.vi)

Génère un tableau représentant un motif sinc.



Si la séquence Y représente le **motif sinc**, le VI génère le motif selon la formule suivante :

$$y_i = a \operatorname{sinc}(i\Delta t - d), \text{ pour } i = 0, 1, 2, \dots, n-1$$

où $\operatorname{sinc}(x) = \frac{\sin(\pi x)}{\pi x}$, a représente l'**amplitude**, Δt représente l'intervalle

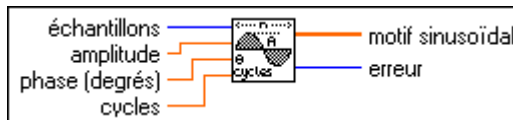
d'échantillonnage **delta t**, d représente le **retard** et n représente le nombre d'**échantillons**.

Le lobe principal de la fonction sinc, $\operatorname{sinc}(x)$, représente la partie de la courbe sinc limitée par la région $-1 \leq x \leq 1$.

Lorsque $|x| = 1$, $\operatorname{sinc}(x) = 0.0$, le pic de la fonction sinc correspond à $x = 0$. En utilisant la règle de l'Hôpital, vous pouvez montrer que $\operatorname{sinc}(0) = 1$ et qu'il s'agit de la valeur de crête. Ainsi, le lobe principal est la région de la courbe sinc située entre le premier jeu de zéros à gauche et à droite de la valeur sinc.

Motif sinusoïdal (Sine Pattern.vi)

Génère un tableau représentant un motif sinusoïdal.



Si la séquence Y représente le **Motif sinusoïdal**, le VI génère le motif selon la formule suivante :

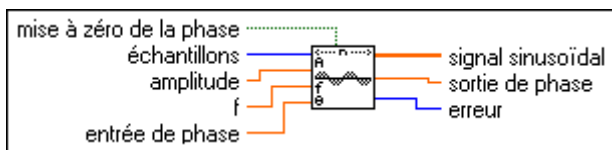
$$y_i = a \sin(x_i), \text{ pour } i = 0, 1, 2, \dots, n-1$$

où $x_i = \frac{2\pi i k}{n} + \frac{\pi\phi_0}{180}$, a représente l'**amplitude**, k représente le nombre de **cycles** du motif,

ϕ_0 représente la **phase** initiale (**degrés**) et n représente le nombre d'**échantillons**.

Signal sinusoïdal (Sine Wave.vi)

Génère un tableau représentant un signal sinusoïdal.



Si la séquence Y représente le **signal sinusoïdal**, le VI génère le motif selon la formule suivante :

$$y_i = a * \sin(\text{phase}[i]), \text{ pour } i = 0, 1, 2, \dots, n-1$$

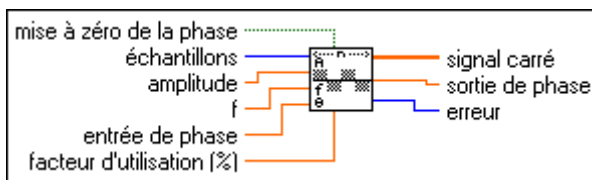
où a représente l'**amplitude** et $\text{phase}[i] = \text{phase_initiale} + f * 360.0 * i$; f représente la fréquence en unités normalisées de cycles/échantillon ; phase_initiale représente l'**entrée de phase** si **mise à zéro de la phase** est vraie ; ou la **sortie de phase** de l'exécution précédente de cette instance du VI si **mise à zéro de la phase** est fausse.

Le VI est ré-entrant et vous pouvez donc l'utiliser pour simuler une acquisition en continu d'un générateur de fonction de signal sinusoïdal. Si le contrôle d'entrée **mise à zéro de la phase** est faux, les appels ultérieurs à une instance spécifique du VI produisent le tableau de sortie **signal sinusoïdal** comprenant les **échantillons** suivants du signal sinusoïdal.

sortie de phase est réglée sur $\text{phase}[n]$, et ce VI ré-entrant utilise cette valeur comme nouvelle **entrée de phase** si **mise à zéro de la phase** est fausse la prochaine fois que le VI s'exécute.

Signal carré (Square Wave.vi)

Génère un tableau représentant un signal carré.



Si la séquence Y représente le **Signal carré**, le VI génère le motif selon la formule suivante :

$$y_i = a * \text{carré}(\text{phase}[i]), \text{ pour } i = 0, 1, 2, \dots, n-1$$

où a représente l'**amplitude** ; n représente le nombre d'**échantillons** ;

$$\text{carré}(\text{phase}[i]) = \begin{cases} 1.0 & 0 \leq p < \left(\frac{\text{fact. d'util.}}{100} \cdot 360 \right) \\ -1.0 & \left(\frac{\text{fact. d'util.}}{100} \cdot 360 \right) \leq p < 360 \end{cases}$$

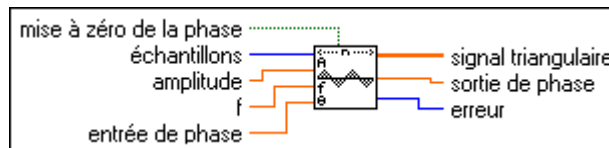
où $p = \text{phase}[i]$ modulo 360.0, $\text{phase}[i] = \text{phase_initiale} + \mathbf{f} * 360.0 * i$; \mathbf{f} représente la fréquence en unités normalisées de cycles/échantillon, phase_initiale représente l'**entrée de phase** si **mise à zéro de la phase** est vraie ; ou la **sortie de phase** de l'exécution précédente de cette instance du VI si **mise à zéro de la phase** est fausse.

Le VI est ré-entrant, et vous pouvez donc l'utiliser pour simuler une acquisition en continu d'un générateur de fonction de signal carré. Si la commande d'entrée **mise à zéro de la phase** est fausse, les appels ultérieurs à une instance spécifique de ce VI produisent un tableau de sortie **Signal carré** comprenant les **échantillons** suivants du signal carré.

Le paramètre **sortie de phase** est réglé sur $\text{phase}[n]$, et ce VI ré-entrant utilise cette valeur comme nouvelle **entrée de phase** si **mise à zéro de la phase** est fausse la prochaine fois que le VI s'exécute.

Signal triangulaire (Triangle Wave.vi)

Génère un tableau représentant un signal triangulaire.



Si la séquence Y représente le **signal triangulaire**, le VI génère le motif selon la formule suivante :

$$y_i = a * \text{tri}(\text{phase}[i]), \text{ pour } i = 0, 1, 2, \dots, n-1$$

où a représente l'**amplitude** ; n représente le nombre d'**échantillons** ;

$$\text{tri}(\text{phase}[i]) = \begin{cases} \frac{p}{90} & 0 \leq p < 90 \\ 2 - \frac{p}{90} & 90 \leq p < 270 \\ \frac{p}{90} + 4 & 270 \leq p < 360 \end{cases}$$

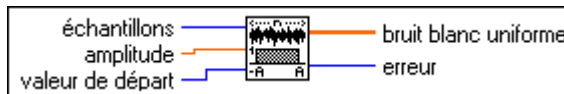
où $p = (\text{phase}[i] \text{ modulo } 360.0)$; $\text{phase}[i] = \text{phase_initiale} + \mathbf{f} * 360.0 * i$; \mathbf{f} représente la fréquence en unités normalisées de cycles/échantillon ; phase_initiale représente l'**entrée de phase** si **mise à zéro de la phase** est vraie ; ou la **sortie de phase** de l'exécution précédente de cette instance du VI si **mise à zéro de la phase** est fausse.

Le VI est ré-entrant et vous pouvez donc l'utiliser pour simuler une acquisition en continu d'un générateur de fonction de signal triangulaire. Si la commande d'entrée **mise à zéro de la phase** est fausse, les appels ultérieurs à une instance spécifique du VI produisent un tableau de sortie **signal triangulaire** comprenant les **échantillons** suivants du signal triangulaire.

sortie de phase est réglée sur $\text{phase}[n]$, et ce VI ré-entrant utilise cette valeur comme nouvelle **entrée de phase** si **mise à zéro de la phase** est fausse la prochaine fois que le VI s'exécute.

Bruit blanc uniforme (Uniform White Noise.vi)

Génère un motif pseudo-aléatoire de distribution uniforme dont la valeur est comprise dans la gamme $[-a:a]$, où a représente la valeur absolue d'**amplitude**.



Le VI génère la séquence pseudo-aléatoire en utilisant une version modifiée de l'algorithme de génération de nombres aléatoires Cycle-très-long. Etant donné que la fonction de densité de probabilité, $f(x)$, du **bruit blanc uniforme** de distribution uniforme est :

$$f(x) = \begin{cases} \frac{1}{2a} & \text{si } -a \leq x \leq a \\ 0 & \text{ailleurs} \end{cases}$$

où a représente la valeur absolue de l'**amplitude** spécifiée, et dont vous pouvez calculer les valeurs attendues, $E\{\bullet\}$, en utilisant la formule

$$E(x) = \int_{-\infty}^{\infty} x(f(x))dx$$

alors la valeur moyenne attendue, μ , et la valeur écart-type attendue, σ , de la séquence pseudo-aléatoire sont

$$\mu = E\{x\} = 0$$

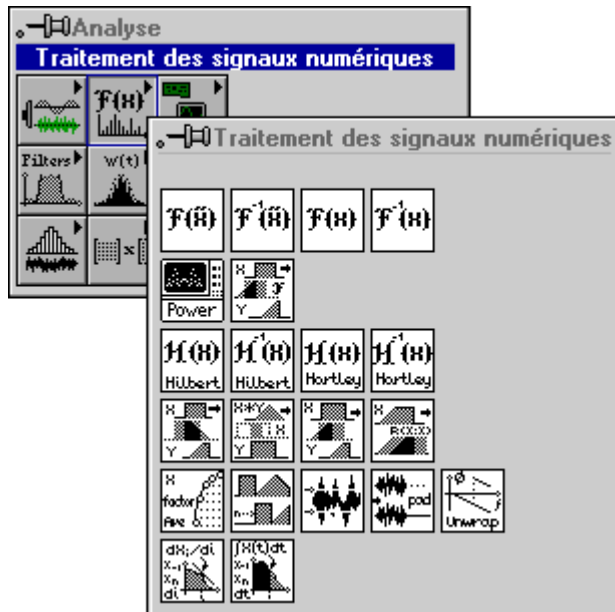
$$\sigma = [E\{(x - \mu)^2\}]^{1/2} = \frac{a}{\sqrt{3}} \approx 0.57735a$$

La séquence pseudo-aléatoire produit approximativement 2^{90} échantillons avant que le motif ne se répète.

VIs de traitement des signaux numériques

Ce chapitre décrit les VIs utilisés pour traiter ou analyser un signal acquis ou simulé. Les VIs de traitement des signaux numériques effectuent des transformées de domaine fréquentiel, des analyses de domaine fréquentiel, des analyses de domaine temporel et d'autres transformées, comme celles de Fourier, Hartley et Hilbert.

Pour ouvrir la palette **Traitement des signaux numériques**, sélectionnez **Fonctions»Analyse»Traitement des signaux numériques**. L'illustration ci-après montre les options disponibles dans la palette **Traitement des signaux numériques**.



Pour des exemples sur l'emploi des VIs de traitement des signaux numériques, reportez-vous aux exemples dans la bibliothèque `examples\analysis\dspxml.11b`.

Description des VIs de traitement de signaux

Les VIs suivants de traitement de signaux sont disponibles.

Autocorrélation (AutoCorrelation.vi)

Calcule l'autocorrélation de la séquence d'entrée **X**.



L'autocorrélation $R_{xx}(t)$ d'une fonction $x(t)$ est définie comme

$$R_{xx}(t) = x(t) \otimes x(t) = \int_{-\infty}^{\infty} x(\tau)x(t + \tau)d\tau$$

où le symbole \otimes indique une corrélation.

Pour l'implémentation discrète de ce VI, considérez que Y représente une séquence dont l'indexage peut être négatif, laissez n comme nombre d'éléments dans la séquence d'entrée **X**, et supposez que la valeur des éléments indexés de **X** qui se trouvent hors gamme est égale à zéro,

$$x_j = 0, \quad j < 0 \quad \text{ou} \quad j \geq n$$

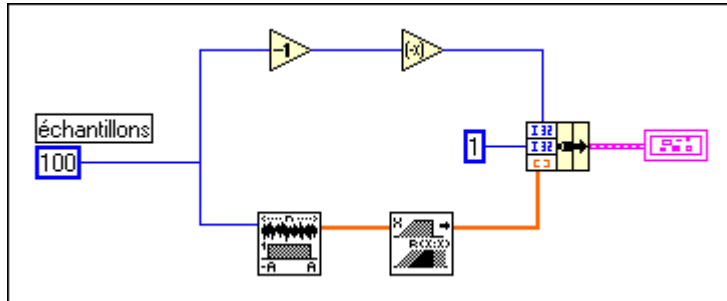
Le VI obtient alors les éléments de Y en utilisant

$$y_j = \sum_{k=0}^{n-1} x_k x_{j+k} \quad \text{pour } j = -(n-1), -(n-2), \dots, -2, -1, 0, 1, 2, \dots, n-1$$

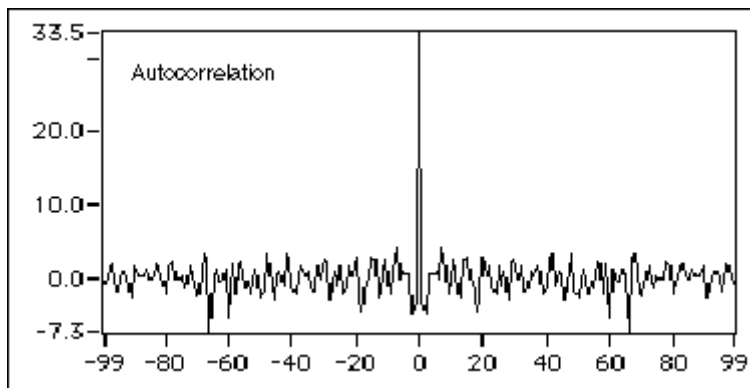
Les éléments de la séquence de sortie **Rxx** sont liés aux éléments dans la séquence Y par :

$$Rxx_i = y_{i-(n-1)} \quad \text{pour } i = 0, 1, 2, \dots, 2n-2$$

Notez que le nombre d'éléments dans la séquence de sortie **Rxx** est $2n - 1$. Vu que vous ne pouvez pas utiliser des nombres négatifs pour indexer des tableaux LabVIEW, la valeur de corrélation correspondant à $t = 0$ est le n^{eme} élément de la séquence de sortie **Rxx**. Ainsi, **Rxx** représente les valeurs de corrélation que le VI a décalé n fois pendant l'indexage. Le diagramme suivant montre une façon d'afficher l'indexage correct pour la fonction d'autocorrélation.



Le graphe suivant est le résultat du diagramme précédent.



FTT complexe (Complex FFT.vi)

Calcule la transformée de Fourier de la séquence d'entrée X .



Vous pouvez utiliser ce VI pour effectuer une FFT sur un tableau de représentations numériques complexes.

Si Y représente la séquence de sortie complexe,

$$Y = F\{X\}$$

Vous pouvez également utiliser ce VI pour effectuer les opérations suivantes quand X est constitué d'un des types de données complexes de LabVIEW.

- La FFT d'une séquence à valeur complexe X
- La DFT d'une séquence à valeur complexe X

Ce VI commence par analyser les données d'entrée et, selon cette analyse, calcule ensuite la transformée de Fourier des données en exécutant une des options précédentes. Toutes ces routines tirent parti des capacités de traitement concurrent de l'unité centrale et du coprocesseur à virgule flottante.

Lorsque le nombre d'échantillons dans la séquence d'entrée X est une puissance valide de 2,

$$n = 2^m \text{ pour } m = 1, 2, 3, \dots, 23$$

où n représente le nombre d'échantillons, le VI calcule la transformée de Fourier rapide en appliquant l'algorithme à base double. La plus grande FFT complexe que le VI puisse calculer est $2^{23} = 8\,388\,608$ (8M).

Lorsque le nombre d'échantillons dans la séquence d'entrée X n'est pas une puissance valide de 2,

$$n \neq 2^m \text{ pour } m = 1, 2, 3, \dots, 23$$

où n représente le nombre d'échantillons, le VI calcule la transformée de Fourier discrète en appliquant l'algorithme "chirp-z". La plus grande DFT complexe qui puisse être calculée est $2^{22} - 1 = 4\,194\,303$ (4M - 1).



Remarque *Etant donné que le VI effectue la transformée intégrée, la FFT est avantageuse sur le plan de la vitesse et de l'efficacité de la mémoire. La taille de la séquence d'entrée, cependant, doit être une puissance de 2. La DFT peut correctement traiter des séquences de toutes tailles, mais la DFT est plus lente que la FFT et utilise plus de mémoire, car elle doit stocker les résultats intermédiaires pendant le traitement.*

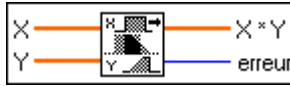
Y étant la séquence de sortie complexe et n son nombre d'échantillons, il est possible de montrer que

$$Y_{n-i} = Y_{-i}$$

Ce qui signifie que vous pouvez interpréter le $(n - i)$ ^{ème} élément de Y comme $-i$ ^{ème} élément de la séquence, si sa réalisation physique était possible, qui représente le i ^{ème} harmonique négatif.

Convolution

Calcule la convolution de la séquence d'entrées **X** et **Y**.



La convolution $h(t)$ des signaux $x(t)$ et $y(t)$ est définie comme

$$h(t) = x(t) * y(t) = \int_{-\infty}^{\infty} x(\tau)y(t - \tau)d\tau$$

où le symbole $*$ indique une convolution.

Pour l'implémentation discrète de la convolution, considérez que h représente la séquence de sortie $\mathbf{X} * \mathbf{Y}$, définissez n comme nombre d'éléments dans la séquence d'entrée \mathbf{X} , et m comme le nombre d'éléments dans la séquence d'entrée \mathbf{Y} . En supposant que les éléments indexés de \mathbf{X} et \mathbf{Y} qui se trouvent hors de leur gamme sont zéro,

$$x_i = 0, \quad i < 0 \quad \text{ou} \quad i \geq n$$

et

$$y_j = 0, \quad j < 0 \quad \text{ou} \quad j \geq m$$

vous obtenez alors les éléments de h en utilisant

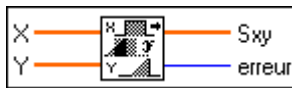
$$h_i = \sum_{k=0}^{n-1} x_k y_{i-k} \quad \text{pour } i = 0, 1, 2, \dots, \text{taille} - 1$$

taille = $n + m - 1$,

où la taille représente le nombre total d'éléments dans la séquence de sortie $\mathbf{X} * \mathbf{Y}$.

Puissance croisée (Cross Power.vi)

Calcule le spectre de puissance croisée des séquences d'entrées **X** et **Y**.



La puissance croisée, $S_{xy}(f)$, des signaux $x(t)$ et $y(t)$ est définie comme

$$S_{xy}(f) = X^*(f)Y(f)$$

où $X^*(f)$ est le conjugué complexe de $X(f)$, $X(f) = F\{x(t)\}$, et $Y(f) = F\{y(t)\}$.

Ce VI utilise la routine FFT ou DFT pour calculer le spectre de puissance croisée, qui est fourni par

$$S_{xy} = \frac{1}{n^2} F^*\{X\} F\{Y\}$$

où S_{xy} représente la séquence de sortie complexe **Sxy** et n représente le nombre d'échantillons qui peuvent recevoir les deux séquences d'entrées **X** et **Y**.

La plus grande puissance croisée que le VI puisse calculer par le biais de la FFT est 2^{23} (8 388 608 ou 8M).

Quand le nombre d'échantillons dans **X** et **Y** sont égaux et sont une puissance valide de 2,

$$n = m = 2^k \text{ pour } k = 1, 2, 3, \dots, 23$$

où n représente le nombre d'échantillons dans **X** et m représente le nombre d'échantillons dans **Y**, le VI appelle directement la routine FFT pour calculer la séquence complexe de puissance croisée. Cette méthode est extrêmement utile en ce qui concerne le temps d'exécution et la gestion de mémoire car le VI effectue les opérations intégrées.

Quand le nombre d'échantillons dans **X** et **Y** ne sont pas égaux,

$$n \neq m$$

où n représente le nombre d'échantillons dans **X** et m représente le nombre d'échantillons dans **Y**, le VI commence par redimensionner la plus petite séquence en la remplissant avec des zéros jusqu'à ce qu'elle corresponde à la taille de la séquence plus large. Si cette taille est une puissance valide de 2,

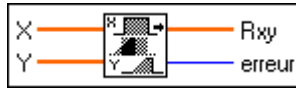
$$\max(n,m) = 2^k \text{ pour } k = 1, 2, 3, \dots, 23$$

le VI calcule le spectre de puissance croisée en utilisant la FFT ; sinon le VI utilise la DFT plus lente pour calculer le spectre de puissance croisée. Ainsi, la taille de la séquence de sortie complexe est

$$\text{taille} = \max(n, m)$$

Corrélation croisée (CrossCorrelation.vi)

Calcule la corrélation croisée de la séquence d'entrées **X** et **Y**.



La corrélation croisée $R_{xy}(t)$ des signaux $x(t)$ et $y(t)$ est définie comme

$$R_{xy}(t) = x(t) \otimes y(t) = \int_{-\infty}^{\infty} x(\tau)y(t + \tau)d\tau$$

où le symbole \otimes indique une corrélation.

Pour l'implémentation discrète de ce VI, considérez que h représente une séquence dont l'indexage peut être négatif, définissez n comme nombre d'éléments dans la séquence d'entrée **X**, m comme nombre d'éléments dans la séquence **Y**, et supposez que les éléments indexés de **X** et **Y** qui se trouvent hors gamme sont égaux à zéro,

$$x_j = 0, \quad j < 0 \quad \text{ou} \quad j \geq n$$

et

$$y_j = 0, \quad j < 0 \quad \text{ou} \quad j \geq m$$

Le VI obtient alors les éléments de h en utilisant

$$h_j = \sum_{k=0}^{n-1} x_k y_{j+k} \quad \text{pour } j = -(n-1), -(n-2), \dots, -2, -1, 0, 1, 2, \dots, m-1$$

Les éléments de la séquence de sortie **Rxy** sont liés aux éléments de la séquence h par

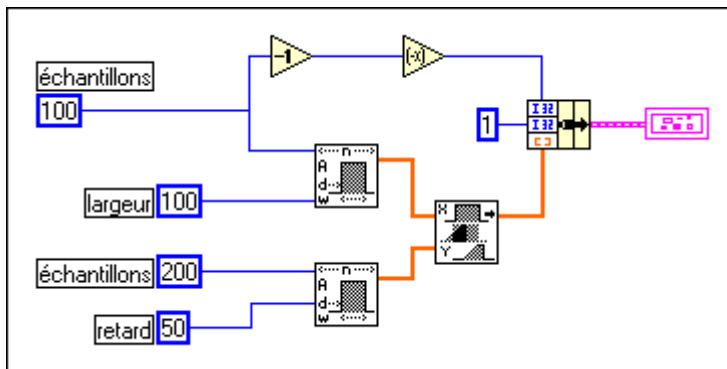
$$Rxy_i = h_{i-(n-1)} \quad \text{pour } i = 0, 1, 2, \dots, \text{taille}-1$$

$$\text{taille} = n + m - 1$$

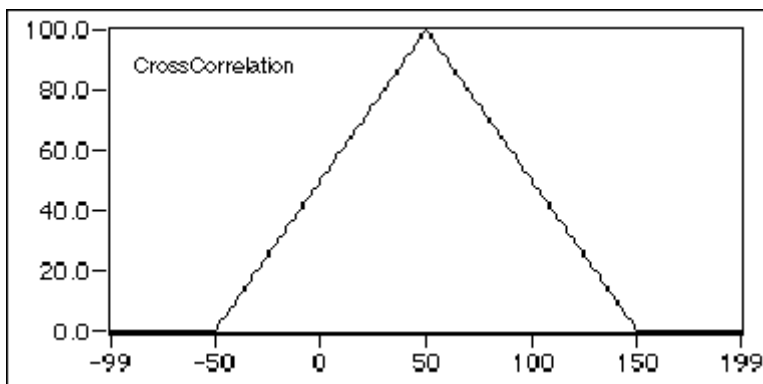
où la taille est le nombre d'éléments dans la séquence de sortie **Rxy**.

Etant donné que vous ne pouvez pas indexer des tableaux LabVIEW avec des nombres négatifs, la valeur de corrélation croisée correspondant à $t = 0$ est le $n^{\text{ème}}$ élément de la séquence de sortie **Rxy**. Ainsi, **Rxy** représente les valeurs de corrélation que le VI a déplacées n fois dans l'indexage.

Le diagramme suivant montre une façon d'indexer le VI "Corrélation croisée" (CrossCorrelation.vi).



Le graphe suivant est le résultat du diagramme précédent.



Décimer (Decimate.vi)

Décime la séquence d'entrée **X** par le **facteur de décimation** et la commande binaire de **moyennage**.



Si Y représente la séquence de sortie **Tableau décimé**, le VI obtient les éléments de la séquence Y en utilisant

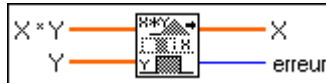
$$y_i = \begin{cases} x_{im} & \text{si moy. est faux} \\ \frac{1}{m} \sum_{k=0}^{m-1} x_{(im+k)} & \text{si moy. est vrai} \end{cases} \quad \text{pour } i = 0, 1, 2, \dots, \text{taille}-1$$

$$\text{taille} = \text{tronc}\left(\frac{n}{m}\right)$$

où n représente le nombre d'éléments de **X**, m représente le **facteur de décimation**, *moy.* représente l'option de **moyennage** et *taille* représente le nombre d'éléments de la séquence de sortie **Tableau décimé**.

Déconvolution (Deconvolution.vi)

Calcule la déconvolution de la séquence d'entrées **X * Y** et **Y**.



Le VI peut employer les identités de Fourier pour effectuer l'opération de convolution car

$$x(t) * y(t) \Leftrightarrow X(f) Y(f)$$

est une paire de transformées de Fourier, où le symbole $*$ indique une convolution et la déconvolution est l'inverse de l'opération de convolution. Si $h(t)$ est le signal issu de la déconvolution des signaux $x(t)$ et $y(t)$, le VI obtient $h(t)$ en utilisant l'équation

$$h(t) = F^{-1}\left(\frac{X(f)}{Y(f)}\right)$$

où $X(f)$ représente la transformée de Fourier de $x(t)$, et $Y(f)$ représente la transformée de Fourier de $y(t)$.

Le VI effectue l'implémentation discrète de la déconvolution en utilisant les étapes suivantes :

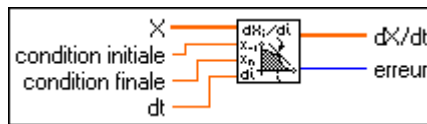
1. Calcule la transformée de Fourier de la séquence d'entrée $\mathbf{X} * \mathbf{Y}$.
2. Calcule la transformée de Fourier de la séquence d'entrée \mathbf{Y} .
3. Divise la transformée de Fourier de $\mathbf{X} * \mathbf{Y}$ par la transformée de Fourier de \mathbf{Y} . Nomme la nouvelle séquence H .
4. Calcule la transformée de Fourier inverse of H pour obtenir la séquence déconvolutionnée \mathbf{X} .



Remarque *L'opération de déconvolution est une opération numériquement instable, et il n'est pas toujours possible de résoudre le système de façon numérique. Le calcul de la déconvolution par le biais des FFT est peut-être l'algorithme générique le plus stable qui n'exige pas de techniques DSP avancées. Cependant, il n'est pas sans erreur (en particulier s'il existe des zéros dans la transformée de Fourier de la séquence d'entrée Y).*

Dérivée $x(t)$ (Derivative $x(t)$.vi)

Effectue une différenciation discrète du signal échantillonné \mathbf{X} .



La différenciation $f(t)$ d'une fonction $F(t)$ est définie comme

$$f(t) = \frac{d}{dt}F(t)$$

Supposez que Y représente la séquence de sortie échantillonnée $\mathbf{dX/dt}$. L'implémentation discrète est fournie par

$$y_i = \frac{1}{2dt}(x_{i+1} - x_{i-1}) \text{ pour } i = 0, 1, 2, \dots, n-1$$

où n représente le nombre d'échantillons de $\mathbf{x(t)}$,

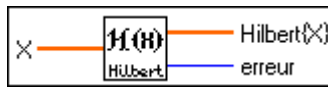
x_{-1} est spécifié par la **condition initiale** quand $i = 0$ et

x_n est spécifié par la **condition finale** quand $i = n - 1$.

La **condition initiale** et la **condition finale** minimisent les erreurs de limites.

Transformée de Hilbert rapide (Fast Hilbert Transform.vi)

Calcule la transformée de Hilbert rapide de la séquence d'entrée **X**.



La transformée de Hilbert d'une fonction $x(t)$ est définie comme

$$h(t) = H\{x(t)\} = -\frac{1}{\pi} \int_{-\infty}^{\infty} \frac{x(\tau)}{t - \tau} d\tau$$

En utilisant des identités de Fourier, vous pouvez montrer que la transformée de Fourier de la transformée de Hilbert de $x(t)$ est

$$h(t) \Leftrightarrow H(f) = -j \operatorname{sgn}(f) X(f)$$


où $x(t) \Leftrightarrow X(f)$ est une paire de transformées de Fourier et

$$\operatorname{sgn}(f) = \begin{cases} 1 & f > 0 \\ 0 & f = 0 \\ -1 & f < 0 \end{cases}$$

Le VI exécute les étapes suivantes pour effectuer l'implémentation discrète de la transformée de Hilbert à l'aide de routines FFT basées sur la paire de transformées de Fourier $h(t) \Leftrightarrow H(f)$ (consultez le format de sortie du VI FFT pour plus d'informations) :

1. Exécuter la transformée de Fourier de la séquence d'entrée **X** : $Y = F\{X\}$.
2. Définir la composante DC sur zéro : $Y_0 = 0$.
3. Si la séquence Y représente une taille paire, définir la composante Nyquist sur zéro : $Y_{\text{Nyq}} = 0$.
4. Multiplier les harmoniques positives par $-j$.
5. Multiplier les harmoniques négatives par j . Nommer la nouvelle séquence H , qui représente la forme $H_k = -j \operatorname{sgn}(k) Y_k$.
6. Calculer la transformée de Fourier inverse de H pour obtenir la transformée de Hilbert de **X**.

Utilisez la transformée de Hilbert pour extraire des informations instantanées sur la phase, obtenir l'enveloppe d'un signal oscillant, obtenir des spectres de bande latérale unique, détecter des échos et réduire les taux d'échantillonnage.

 **Remarque** *Etant donné que le VI règle les composantes DC et Nyquist sur zéro quand le nombre d'éléments dans la séquence d'entrée est pair, vous ne pouvez pas toujours récupérer le signal d'origine avec une transformée de Hilbert inverse. La transformée de Hilbert est utile avec les signaux à bande passante limités, qui excluent les composantes DC et Nyquist.*

FHT

Calcule la transformée de Hartley rapide (FHT) de la séquence d'entrée **X**.



La transformée de Hartley d'une fonction $x(t)$ est définie comme

$$X(f) = \int_{-\infty}^{\infty} x(t) \text{cas}(2\pi ft) dt$$

où $\text{cas}(x) = \cos(x) + \sin(x)$.

Si Y représente la séquence de sortie **Hartley{X}** obtenue par le biais de FHT, alors Y est obtenue grâce à l'implémentation discrète de l'intégrale de Hartley :

$$Y_k = \sum_{i=0}^{n-1} X_i \text{cas}\left(\frac{2\pi ik}{n}\right), \text{ pour } k = 0, 1, 2, \dots, n-1$$

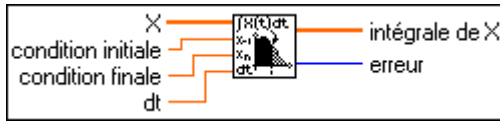
où n représente le nombre d'éléments de X .

La FHT mappe les séquences de valeurs réelles en séquences de valeurs réelles de domaine fréquentiel. Vous pouvez l'utiliser au lieu de la transformée de Fourier pour convolutionner, déconvolutionner et corrélérer des signaux et pour trouver le spectre de puissance. Vous pouvez également dériver la transformée de Fourier à partir de la transformée de Hartley.

Quand les séquences à traiter sont des séquences de valeurs réelles, la transformée de Fourier produit des séquences de valeurs complexes dont la moitié des informations sont redondantes. L'avantage de la transformée FHT sur la FFT est que la FHT utilise moitié moins de mémoire pour produire les mêmes informations que la FFT. De plus, la FHT est calculée de manière intégrée et est toute aussi efficace que la FFT. L'inconvénient de la FHT est que la taille de la séquence d'entrée doit être une puissance valide de 2.

Intégrale x(t) (Integral x(t).vi)

Effectue l'intégration discrète du signal échantillonné **X**.



L'intégrale $F(t)$ d'une fonction $f(t)$ est définie comme

$$F(t) = \int f(t) dt$$

Supposez que Y représente la séquence de sortie échantillonnée **Intégrale X**. Le VI obtient les éléments de Y en utilisant

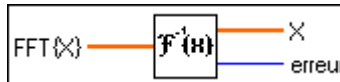
$$y_i = \frac{1}{6} \sum_{j=0}^i (x_{j-1} + 4x_j + x_{j+1}) dt \quad \text{pour } i = 0, 1, 2, \dots, n-1$$

où n représente le nombre d'éléments de X , x_{-1} est spécifié par la **condition initiale** quand $i = 0$, et x_n est spécifié par la **condition finale** quand $i = n - 1$.

La **condition initiale** et la **condition finale** minimisent les erreurs générales en augmentant la précision aux limites, en particulier quand le nombre d'échantillons est petit.

FFT complexe inverse (Inverse Complex FFT.vi)

Calcule la transformée de Fourier inverse de la séquence d'entrée complexe **FFT {X}**.



Vous pouvez utiliser ce VI pour effectuer une FFT inverse sur un tableau exprimée dans une représentation numérique complexe de LabVIEW.

Si Y représente la séquence de sortie, alors

$$Y = F^{-1}\{X\}$$

Vous pouvez utiliser ce VI pour effectuer les opérations suivantes quand **FFT {X}** est exprimé dans un des types de données complexes de LabVIEW :

- La FFT inverse d'une séquence de valeur complexe X
- La DFT inverse d'une séquence de valeur complexe X

Ce VI commence par analyser les données d'entrée et, selon cette analyse, calcule leur transformée inverse de Fourier en exécutant l'une des options précédentes. Toutes ces routines tirent parti des capacités de traitement concurrent de l'unité centrale et du coprocesseur à virgule flottante.

Quand le nombre d'échantillons dans la séquence d'entrée X est une puissance valide de 2,
 $n = 2^m$ pour $m = 1, 2, 3, \dots, 23$

où n représente le nombre d'échantillons, le VI calcule la FFT inverse en appliquant l'algorithme à base double. Avec une FFT inverse complexe, la séquence la plus longue que le VI puisse calculer est $2^{23} = 8\,388\,608$ (8M).

Quand le nombre d'échantillons dans la séquence d'entrée X n'est pas une puissance valide de 2,

$n \neq 2^m$ pour $m = 1, 2, 3, \dots, 23$

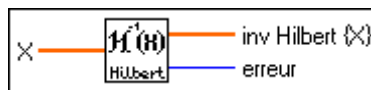
où n représente le nombre d'échantillons, le VI calcule la DFT inverse en appliquant l'algorithme "chirp-z". Avec une DFT inverse complexe, la séquence la plus longue que le VI puisse calculer est $2^{22} - 1$ (4 194 303 ou $4M - 1$).



Remarque *Etant donné que le VI effectue la transformée de manière intégrée, la FFT est avantageuse sur le plan de la vitesse et de l'efficacité de la mémoire. La taille de la séquence d'entrée, cependant, doit être une puissance de 2. La DFT peut correctement traiter des séquences de toutes tailles, mais la DFT est plus lente que la FFT et utilise plus de mémoire, car elle doit stocker les résultats intermédiaires pendant le traitement.*

Transformée de Hilbert rapide inverse (Inverse Fast Hilbert Transform.vi)

Calcule la transformée de Hilbert inverse rapide de la séquence d'entrée X .



La transformée de Hilbert inverse d'une fonction $h(t)$ est définie comme

$$h(t) = H^{-1}\{h(t)\} = \frac{1}{\pi} \int_{-\infty}^{\infty} \frac{h(\tau)}{t - \tau} d\tau$$

en utilisant la définition de la transformée de Hilbert

$$h(t) = H\{x(t)\} = -\frac{1}{\pi} \int_{-\infty}^{\infty} \frac{x(\tau)}{t - \tau} d\tau$$

Vous obtenez la transformée de Hilbert inverse en effectuant une opération de négation de la transformée de Hilbert directe

$$x(t) = H^{-1}\{h(t)\} = -H\{h(t)\}$$

Le VI exécute les étapes suivantes pour effectuer l'implémentation discrète de la transformée de Hilbert inverse à l'aide de la transformée de Hilbert.

1. Effectuer la transformée de Hilbert de la séquence d'entrée \mathbf{X} : $Y = H\{X\}$.
2. Effectuer la négation de Y pour obtenir la transformée de Hilbert inverse : $H^{-1}\{X\} = -Y$.

FHT inverse (Inverse FHT.vi)

Calcule la transformée de Hartley inverse rapide (FHT) de la séquence d'entrée \mathbf{X} .



La transformée de Hartley inverse d'une fonction $X(f)$ est définie comme

$$x(t) = \int_{-\infty}^{\infty} X(f) \text{cas}(2\pi ft) df$$

où $\text{cas}(x) = \cos(x) + \sin(x)$.

Si Y représente la séquence de sortie **Inv FHT {X}**, le VI calcule Y par le biais de l'implémentation discrète de l'intégrale de Hartley inverse :

$$Y_k = \frac{1}{n} \sum_{i=0}^{n-1} X_i \cos \frac{2\pi ik}{n}, \text{ pour } k = 0, 1, 2, \dots, n-1$$

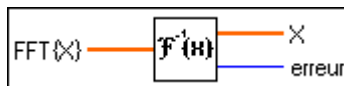
où n représente le nombre d'éléments de \mathbf{X} .

La transformée de Hartley inverse mappe des séquences de valeurs réelles de fréquence en séquences de valeurs réelles. Vous pouvez l'utiliser au lieu de la transformée de Fourier pour convolutionner, déconvolutionner et corrélérer des signaux et pour trouver le spectre de puissance. Vous pouvez également dériver la transformée de Fourier à partir de la transformée de Hartley.

Reportez-vous à la section *FHT* figurant plus haut dans ce chapitre pour comparer les transformées de Fourier et de Hartley.

FFT réelle inverse (Inverse Real FFT.vi)

Calcule la transformée de Fourier réelle inverse rapide (FFT) ou la transformée de Fourier réelle inverse discrète (DFT) de la séquence d'entrée **FFT{X}**.



La séquence d'entrée est à valeur complexe. Ce VI détermine automatiquement les options suivantes :

- FFT réelle inverse d'une séquence à valeur complexe si la taille est une puissance de 2.
- DFT réelle inverse d'une séquence à valeur complexe si la taille n'est pas une puissance de 2.

Ce VI exécute des routines FFT inverse si la taille de la séquence d'entrée est une puissance valide de 2 :

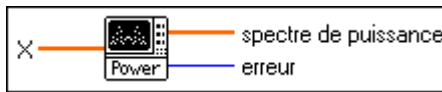
$$\text{taille} = 2^m, m = 1, 2, \dots, 23$$

Si la taille de la séquence d'entrée n'est pas une puissance de 2, ce VI appelle une routine DFT inverse efficace.

La séquence de sortie $\mathbf{X} = \text{FFT réelle inverse} [\mathbf{FFT}\{\mathbf{X}\}]$ est réelle et est retournée dans un tableau réel.

Spectre de puissance (Power Spectrum.vi)

Calcule le spectre de puissance de la séquence d'entrée **X**.



Le **Spectre de puissance** $S_{xx}(f)$ d'une fonction $x(t)$ est défini comme

$$S_{xx}(f) = X^*(f)X(f) = |X(f)|^2$$

où $X(f) = F\{x(t)\}$, et $X^*(f)$ représente le complexe conjugué de $X(f)$.

Ce VI utilise les routines FFT et DFT pour calculer le spectre de puissance, qui est fourni par

$$S_{xx} = \frac{1}{n} |F\{\mathbf{X}\}|^2$$

où S_{xx} représente la séquence de sortie **Spectre de puissance** et n est le nombre d'échantillons de la séquence d'entrée **X**.

Quand le nombre d'échantillons, n , de la séquence d'entrée **X** est une puissance valide de 2, $n = 2^m$ pour $m = 1, 2, 3, \dots, 23$

ce VI calcule la FFT de la séquence à valeur réelle en utilisant l'algorithme à base double et effectue la mise à l'échelle du carré de l'amplitude. Le plus grand spectre de puissance que le VI peut calculer en utilisant la FFT est 2^{23} (8 388 608 ou 8M).

Quand le nombre d'échantillons de la séquence d'entrée X n'est pas une puissance valide de 2, $n \neq 2^m$ pour $m = 1, 2, 3, \dots, 23$

où n représente le nombre d'échantillons, ce VI calcule la transformée de Fourier discrète de la séquence à valeur réelle en utilisant l'algorithme "chirp-z" et effectue la mise à l'échelle du carré de l'amplitude. Le plus grand spectre de puissance que le VI peut calculer en utilisant la DFT rapide est $2^{22} - 1$ (4 194 303 ou 4M - 1).

Le calcul FFT du spectre de puissance est efficace en terme de temps et de mémoire car la transformée est réelle et est exécutée dans le même espace. Cependant, la taille de la séquence d'entrée doit être exactement une puissance de 2. La version DFT calcule de façon efficace le spectre de puissance des séquences de toutes tailles. La version DFT est plus lente que la version FFT, utilise plus de mémoire et sa mise à l'échelle n'est pas aussi efficace.

Supposez que Y soit la transformée de Fourier de la séquence d'entrée \mathbf{X} , et n le nombre d'échantillons. Il est possible de montrer que

$$|Y_{n-i}|^2 = |Y_{-i}|^2$$

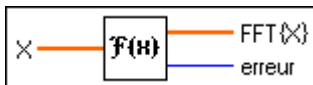
Vous pouvez interpréter la puissance du $(n-i)$ ^{ème} élément de Y comme la puissance du $-i$ ^{ème} élément de la séquence, qui représente la puissance de la i ^{ème} harmonique *négative*. Vous pouvez déterminer la puissance totale de la i ^{ème} harmonique (composantes DC et Nyquist non comprises) en utilisant

$$2|Y_i|^2 = |Y_i|^2 + |Y_{n-i}|^2, \quad 0 < i < \frac{n}{2}$$

La puissance totale des composantes DC et Nyquist sont $|Y_0|^2$ et $|Y_{n/2}|^2$, respectivement.

FFT réelle (Real FFT.vi)

Calcule la transformée de Fourier réelle rapide (FFT) ou la transformée de Fourier réelle discrète (DFT) de la séquence d'entrée \mathbf{X} .



La séquence d'entrée est à valeur réelle. Le VI "FFT réelle" (Real FFT.vi) détermine automatiquement les options suivantes :

- FFT d'une séquence à valeur réelle
- DFT d'une séquence à valeur réelle

Le VI "FFT réelle" (Real FFT.vi) exécute des routines FFT si la taille de la séquence d'entrée est une puissance valide de 2 :

$$\text{taille} = 2m, m = 1, 2, \dots, 23$$

Si la taille de la séquence d'entrée n'est pas une puissance de 2, le VI "FFT réelle" (Real FFT.vi) appelle une routine DFT réelle efficace.

La séquence de sortie $Y = \text{FFT réelle}[\mathbf{X}]$ est complexe et est retournée dans un tableau complexe :

$$Y = Y_{\text{Re}} + jY_{\text{Im}}$$

Lissage de phase (Unwrap Phase.vi)

Lisse le tableau **Phase** en éliminant des discontinuités dont les valeurs absolues dépassent π .



$Y[i] = \text{Clip}\{X[i]\}$

Coupe les éléments du **Tableau d'entrée** conformément aux limites spécifiées par **limite supérieure** et **limite inférieure**.



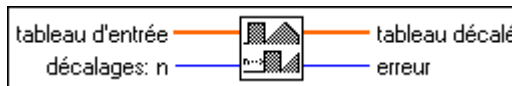
Supposez que la séquence Y représente la séquence de sortie **Tableau coupé** ; les éléments de Y sont alors liés aux éléments du **Tableau d'entrée** par

$$y_i = \begin{cases} a & x_i > a \\ x_i & b \leq x_i \leq a \text{ pour } i = 0, 1, 2, \dots, n-1 \\ b & x_i < b \end{cases}$$

où n représente le nombre d'éléments du **Tableau d'entrée**, a représente la **limite supérieure** et b représente la **limite inférieure**.

$Y[i] = X[i-n]$


Décale les éléments du **Tableau d'entrée** par le nombre spécifié de décalages.



Supposez que la séquence Y représente la séquence de sortie **Tableau décalé** ; les éléments de Y sont alors liés aux éléments de X par

$$y_i = \begin{cases} x_{i-\text{décalages}} & \text{si } 0 \leq i - \text{décalages} < n \text{ pour } i = 0, 1, 2, \dots, n-1, \\ 0 & \text{ailleurs} \end{cases}$$

où n représente le nombre d'éléments du **Tableau d'entrée**.

 **Remarque** *Ce VI ne permute pas les éléments du tableau. Il élimine les éléments de la séquence d'entrée décalée hors du tableau et vous ne pouvez pas les récupérer en décalant le tableau dans le sens opposé.*

Comblé avec des zéros (Zero Padder.vi)

Redimensionne la séquence d'entrée **Tableau d'entrée** à la puissance valide de 2 immédiatement supérieure, définit les nouveaux éléments de la séquence sur zéro et laisse intacts les premiers n éléments, où n représente le nombre d'échantillons de la séquence d'entrée.

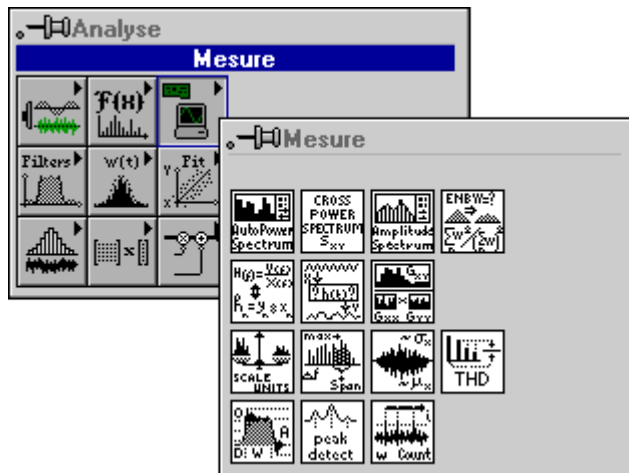


Ce VI est utile quand la taille des buffers de données acquises n'est pas une puissance de 2 et que vous souhaitez tirer parti des algorithmes de traitement rapides dans les VIs d'analyse. Ces algorithmes comprennent des transformées de Fourier, des spectres de puissance et des FHT qui sont extrêmement performants pour les tailles de buffers égales à une puissance de 2.

Vis de mesure

Ce chapitre décrit les VIs de mesure, conçus pour réaliser l'analyse en base DFT et FFT de signaux acquis avec des applications de mesure de fréquence comparables aux instruments classiques de mesure (analyseur du signal dynamique, par exemple).

Pour accéder à la palette **Mesure**, sélectionnez **Fonctions»Analyse»Mesure**. L'illustration suivante présente les options disponibles dans la palette **Mesure**.



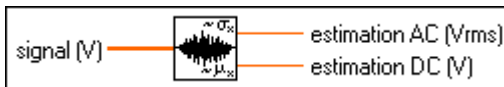
Pour des exemples sur l'utilisation des VIs de mesure, reportez-vous aux exemples concernant l'acquisition de données dans `exemples\analysis\measure\daqmeas.llb` et en utilisant des signaux simulés dans `exemples\analysis\measure\measxmpl.llb`.

Descriptions des VIs de mesure

Les VIs de mesure suivants sont disponibles.

Estimation AC & DC (AC & DC Estimator.vi)

Calcule une estimation des niveaux AC et DC du signal d'entrée.



Spectre d'amplitude et de phase (Amplitude et Phase Spectrum.vi)

Calcule la phase et l'amplitude du spectre d'amplitude unilatéral mis à l'échelle d'un signal du domaine temporel réel.



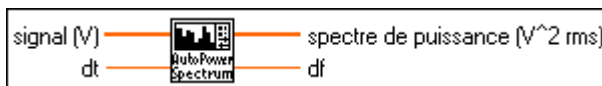
Le VI calcule le spectre d'amplitude comme

$$\frac{\text{FFT}(\text{Signal})}{N}$$

où N représente le nombre de points du tableau **Signal**. Le VI effectue alors la conversion du spectre d'amplitude en spectres unilatéraux de phase et de valeur de moyenne quadratique de l'amplitude.

Spectre de puissance auto (Auto Power Spectrum.vi)

Calcule le spectre de puissance auto unilatéral et mis à l'échelle d'un signal du domaine temporel.



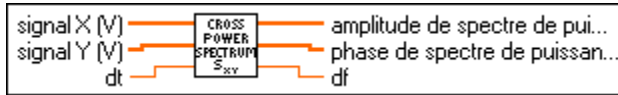
Ce VI calcule le spectre de puissance comme

$$\frac{\text{FFT}^*(\text{Signal}) \times \text{FFT}(\text{Signal})}{N^2}$$

où N représente le nombre de points du tableau **Signal** et $*$ indique un complexe conjugué. Le VI effectue alors la conversion du spectre de puissance en un spectre de puissance unilatéral.

Spectre de puissance croisée (Cross Power Spectrum.vi)

Calcule le spectre de puissance croisée unilatéral, mis à l'échelle, de deux signaux temporels réels. Le spectre de puissance retourne le produit de l'amplitude des signaux X et Y et la différence entre leurs phases (phase Y moins phase X).



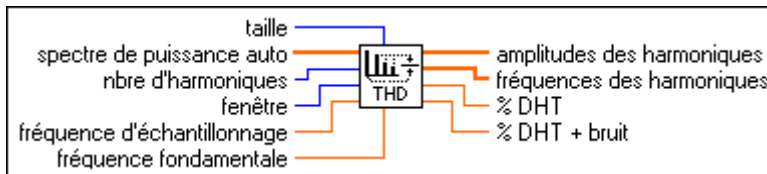
Ce VI calcule le spectre de puissance croisée comme

$$\frac{\text{FFT}*(\text{Signal X}) \times \text{FFT}(\text{Signal Y})}{N^2}$$

où N représente le nombre de points des tableaux **Signal X** ou **Signal Y**. Le VI effectue alors la conversion du spectre de puissance croisée en spectres unilatéraux de phase et d'amplitude.

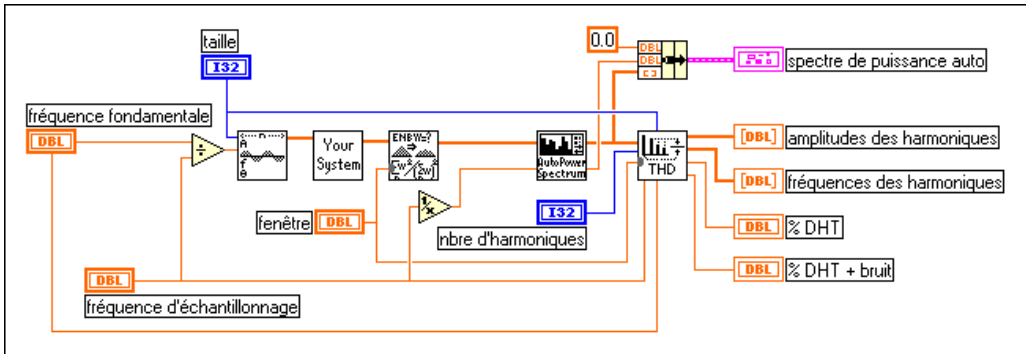
Analyseur harmonique (Harmonic Analyzer.vi)

Détermine les composantes fondamentales et harmoniques (amplitude et fréquence) contenues dans l'entrée **Spectre de puissance auto**, et calcule le pourcentage de la distorsion harmonique totale (%DHT) et la distorsion harmonique totale plus le bruit (%DHT + Bruit).



Vous devez transmettre le spectre de puissance auto-fenêtré de votre signal à ce VI pour un bon fonctionnement. Vous pouvez par exemple transmettre votre signal de domaine temporel à la fenêtre de domaine temporel mise à l'échelle, puis au VI "Spectre de puissance auto" (Auto Power Spectrum.vi), et connecter la sortie du VI "Spectre de puissance auto" (Auto Power Spectrum.vi) à ce VI.

L'illustration suivante présente un exemple de l'utilisation de ce VI.



Réponse impulsionnelle (Impulse Response Function.vi)

Calcule la réponse impulsionnelle d'un réseau en se basant sur des signaux réels X (**Signal de stimulus X**) et Y (**Signal de réponse Y**).



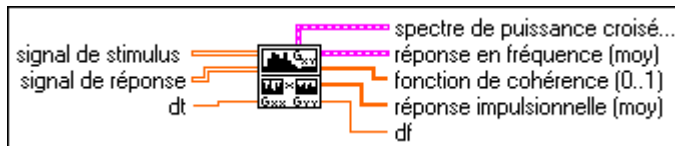
Le paramètre **Réponse impulsionnelle** est exprimé dans le domaine temporel ; vous n'avez donc pas besoin de convertir des unités temporelles en unités de fréquence. Une **Réponse impulsionnelle** représente la transformée inverse d'une fonction de transfert.

Ce VI calcule la **Réponse impulsionnelle** comme

$$FFTInverse \left[\frac{\text{puissance croisée(Stimulus, Réponse)}}{\text{spectre de puissance(Stimulus)}} \right]$$

Fonctions de réseau (moy) (Network Functions (avg).vi)

Calcule plusieurs fonctions de réponse de réseau de deux signaux du domaine temporel réels X (**Signal de stimulus**) et Y (**Signal de réponse**).



Les signaux X (**Signal de stimulus**) et Y (**Signal de réponse**) comprennent la cohérence, la moyenne de la phase et de l'amplitude du spectre de puissance croisée, la moyenne du VI "Fonction de transfert" (Transfer Function.vi) (**Réponse en fréquence**) et la moyenne de **Réponse impulsionnelle**.

Vous calculez normalement ces fonctions à partir du stimulus et du signal de réponses produits par un réseau sous test. La fonction de cohérence montre le contenu fréquentiel du **Signal de réponse** Y produit par **Signal de stimulus** X et mesure la validité de la mesure de la réponse en fréquence du réseau.

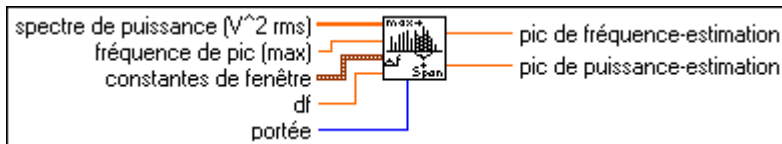
Vous pouvez utiliser ce VI pour mesurer la cohérence entre deux signaux. Le VI réalise la moyenne de plusieurs signaux et stimulus de réponse et obtient des mesures valides de la cohérence. Les paramètres **Spectre de puissance croisée** et **Réponse impulsionnelle** représentent les moyennes quadratiques des VIs portant le même nom. La **Réponse en fréquence** représente la moyenne quadratique des sorties de la réponse en fréquence du VI "Fonction de transfert" (Transfer Function.vi).

Détection de crêtes (Peak Detector.vi)

Pour obtenir des informations sur ce VI, reportez-vous au chapitre 47, *VIs de méthodes numériques supplémentaires*, de ce manuel.

Estimation de puissance & fréq. (Power & Frequency Estimate.vi)

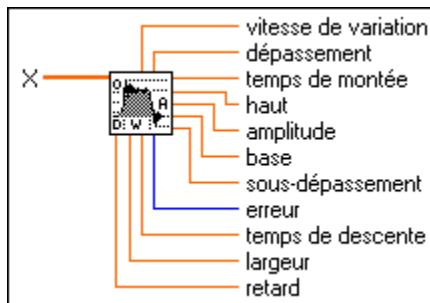
Calcule l'estimation de la puissance et de la fréquence autour d'une crête du spectre de puissance d'un signal du domaine temporel.



Vous pouvez, avec ce VI, obtenir des estimations utiles des fréquences situées entre les lignes de fréquence du spectre. Le VI effectue les corrections nécessaires en fonction de la fenêtre utilisée.

Paramètres d'impulsion (Pulse Parameters.vi)

Analyse la séquence d'entrée **X** pour repérer un motif d'impulsion et détermine quel est le meilleur ensemble de paramètres décrivant l'impulsion.



Les paramètres relatifs à la forme d'onde sont la **vitesse de variation**, le **dépassement**, la ligne supérieure (**haut**), l'**amplitude**, la ligne de base (**base**) et le **sous-dépassement**. Les paramètres relatifs aux temps sont le **temps de montée**, le **temps de descente**, la **largeur** (durée) et le **retard**.

Ce VI réalise les opérations suivantes pour calculer les paramètres de sortie :

1. Trouve les valeurs maximum et minimum dans la séquence d'entrée **X**.
2. Produit l'histogramme de l'impulsion avec une résolution de gamme de 1%.
3. Détermine les modes supérieur et inférieur pour établir les valeurs **haut** et **base**.
4. Détermine le **dépassement**, l'**amplitude** et le **sous-dépassement** à partir des valeurs **haut**, **base**, maximum et minimum.
5. Balaye **X** et détermine les paramètres de **vitesse de variation**, de **temps de montée**, de **temps de descente**, de **largeur** et de **retard**.

Le VI réalise l'interpolation de la **largeur** et du **retard** pour obtenir un résultat plus précis de **largeur** et de **retard**, ainsi que de **vitesse de variation**, le **temps de montée** et le **temps de descente**.

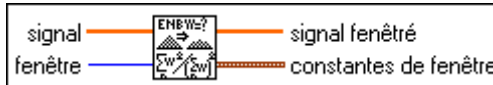
Si **X** comprend un train d'impulsions, le VI utilise le train pour déterminer les paramètres **dépassement**, **haut**, **amplitude**, **base** et **sous-dépassement**. Il n'utilise cependant que la première impulsion du train pour déterminer la **vitesse de variation**, le **temps de montée**, le **temps de descente**, la **largeur** et le **retard**.



Remarque *Etant donné que les impulsions surviennent habituellement en direction négative, ce VI peut différencier les impulsions positives des impulsions négatives et analyser correctement la séquence X. Vous n'avez pas besoin de traiter la séquence avant de l'analyser.*

Fenêtre temporelle mise à l'échelle (Scaled Time Domain Window.vi)

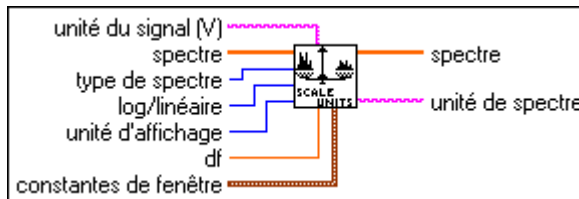
Applique la fenêtre sélectionnée au signal du domaine temporel.



Le VI effectue la mise à l'échelle du résultat de telle façon que lorsque le spectre de puissance ou d'amplitude du **signal fenêtré** est calculé, toutes les fenêtres fournissent le même niveau au sein des contraintes de précision de la fenêtre. Ce VI retourne également les **Constantes de fenêtre** de la fenêtre sélectionnée. Ces constantes sont utiles lorsque vous utilisez des VIs qui réalisent des calculs sur le spectre de puissance, comme les VIs "Estimation de puissance et fréq." (Power & Frequency Estimate.vi) et "Conversion des unités de spectre" (Spectrum Unit Conversion.vi).

Conversion des unités de spectre (Spectrum Unit Conversion.vi)

Convertit le spectre de puissance, d'amplitude ou de gain (rapport d'amplitude) en formats de substitution, dont Log (décibel et dbm) et densité spectrale.

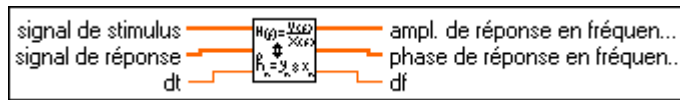


Détection de crêtes avec seuillage (Threshold Peak Detector.vi)

Pour obtenir des informations sur ce VI, reportez-vous au chapitre 47, *VIs de méthodes numériques supplémentaires*, de ce manuel.

Fonction de transfert (Transfer Function.vi)

Calcule la fonction de transfert (également connue comme la réponse en fréquence) à partir du **Signal de stimulus** et du **Signal de réponse** temporels d'un réseau sous test.



Ce VI calcule la fonction de transfert d'un système à partir des signaux réels X (**Signal de stimulus**) et Y (**Signal de réponse**). La sortie représente le gain d'amplitude du réseau, qui ne comporte pas d'unités.

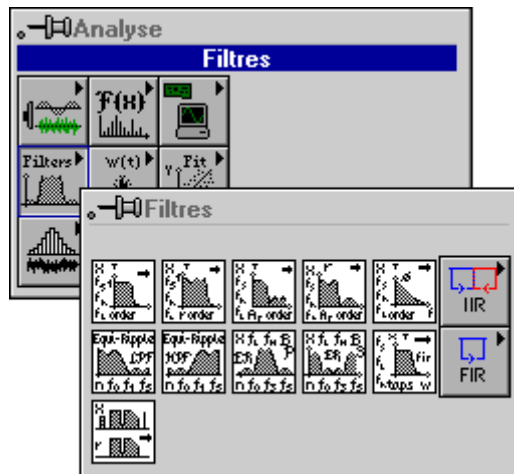
La réponse de fréquence du VI est

$$\frac{\text{Puissance croisée(Stimulus, Réponse)}}{\text{Spectre de puissance(Stimulus)}}$$

VIs Filtres

Ce chapitre décrit les VIs utilisés pour implémenter des filtres RII, RSI et non linéaires.

Pour accéder à la palette **Filtres**, sélectionnez **Fonctions»Analyse»Filtres**. L'illustration suivante présente les options disponibles dans la palette **Filtres**.



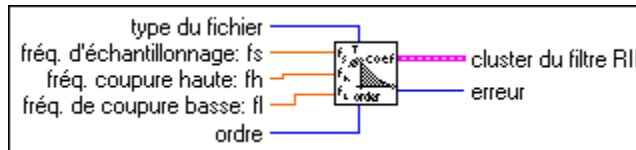
Pour des exemples d'utilisation des VIs Filtres, consultez la bibliothèque `examples\analysis\fltrxmpl.llb`.

Descriptions des VIs Filtres

Les VIs Filtres suivants sont disponibles.

Coefficients de Bessel (Bessel Coefficients.vi)

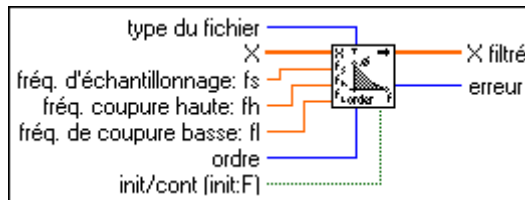
Génère l'ensemble des coefficients de filtre qui implémentent un filtre RII tel que spécifié par le modèle de filtre de Bessel. Vous pouvez ensuite transmettre ces coefficients au VI "Filtre RII cascadié" (IIR Cascade Filter.vi).



Le VI "Coefficients de Bessel" (Bessel Coefficients.vi) est un sous-VI du VI "Filtre de Bessel" (Bessel Filter.vi).

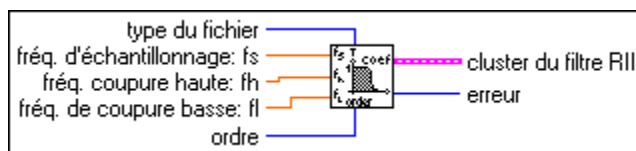
Filtre de Bessel (Bessel Filter.vi)

Génère un filtre de Bessel numérique à partir des paramètres **type de filtre**, **fréq. d'échantillonnage: fs**, **fréq. coupure haute: fh**, **fréq. coupure basse: fl** et **ordre** en appelant le VI "Coefficients de Bessel" (Bessel Coefficients.vi). Le VI appelle ensuite le Filtre RII cascadié pour filtrer une séquence **X** en utilisant ce modèle pour obtenir une séquence **X Filtré** de Bessel.



Coefficients de Butterworth (Butterworth Coefficients.vi)

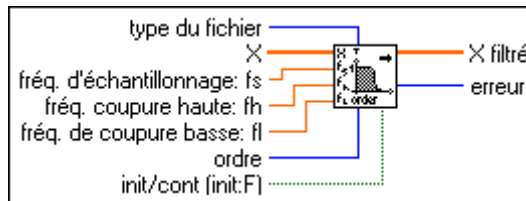
Génère l'ensemble des coefficients de filtre qui implémentent un filtre RII tel que spécifié par le modèle de filtre de Butterworth. Vous pouvez transmettre ces coefficients de filtre (**cluster du filtre RII**) au VI "Filtre RII cascadié" (IIR Cascade Filter.vi) pour filtrer une séquence de données.



Ce VI est un sous-VI du VI “Filtre de Butterworth” (Butterworth Filter.vi).

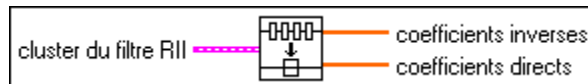
Filtre de Butterworth (Butterworth Filter.vi)

Génère un filtre numérique de Butterworth en utilisant les paramètres **fréq. d'échantillonnage: fs**, **fréq. coupure basse: fl**, **fréq. coupure haute: fh**, **ordre** et **type de filtre** en appelant le VI “Coefficients de Butterworth” (Butterworth Coefficients.vi). Le VI “Filtre de Butterworth” (Butterworth Filter.vi) appelle ensuite le VI “Filtre RII cascadié” (IIR Cascade Filter.vi) pour filtrer une séquence **X** en utilisant ce modèle pour obtenir une séquence **X Filtré** de Butterworth.



Coefficients cascadiés → directs (Cascade → Direct Coefficients.vi)

Convertit des coefficients du filtre RII de la forme cascadiée à la forme directe.



A titre d'exemple, vous pouvez convertir un filtre cascadié, composé de deux étages d'ordre secondaire, en un filtre de forme directe comme suit :

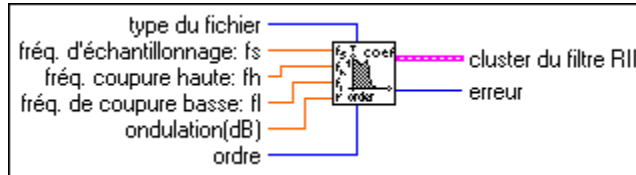
Coefficients inversés : $\{a_{11}, a_{21}, a_{12}, a_{22}\} \rightarrow \{1.0, a_1, a_2, a_3, a_4\}$

Coefficients directs : $\{b_{01}, b_{11}, b_{21}, b_{02}, b_{12}, b_{22}\} \rightarrow \{b_0, b_1, b_2, b_3, b_4\}$

Reportez-vous au VI “Filtre RII cascadié” (IIR Cascade Filter.vi) pour plus d'informations concernant le filtrage en forme cascadiée et au VI “Filtre RII” (IRR Filter.vi) pour plus d'informations sur le filtrage en forme directe.

Coefficients de Chebyshev (Chebyshev Coefficients.vi)

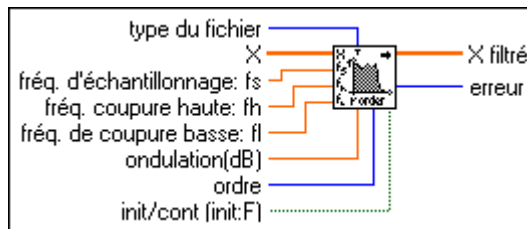
Génère l'ensemble des coefficients de filtre qui implémentent un filtre RII tel que spécifié par le modèle de filtre de Chebyshev. Vous pouvez transmettre ces coefficients au VI "Filtre du cluster RII" (IIR Cluster Filter.vi) pour filtrer une séquence de données.



Le VI "Coefficients de Chebyshev" (Chebyshev Coefficients.vi) est un sous-VI du VI "Filtre de Chebyshev" (Chebyshev Filter.vi).

Filtre de Chebyshev (Chebyshev Filter.vi)

Génère un filtre de Chebyshev numérique en utilisant les paramètres **fréq. d'échantillonnage: fs**, **fréq. coupure basse: fl**, **fréq. coupure haute: fh**, **ondulation**, **ordre** et **type de filtre** en appelant le VI "Coefficients de Chebyshev" (Chebyshev Coefficients.vi). Le VI "Filtre de Chebyshev" (Chebyshev Filter.vi) filtre la séquence **X** et utilise ce modèle pour obtenir une séquence de **X Filtré** de Chebyshev en appelant le VI "Filtre RII cascadié" (IIR Cascade Filter.vi).

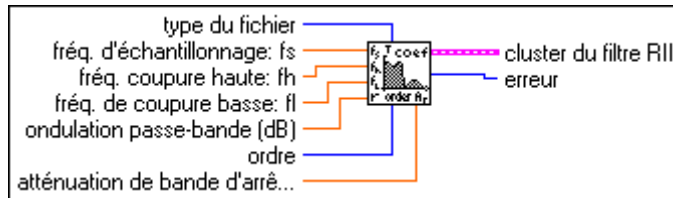


Convolution

Pour des informations relatives au VI Convolution, reportez-vous au chapitre 39, [VIs de traitement des signaux numériques](#).

Coefficients elliptiques (Elliptic Coefficients.vi)

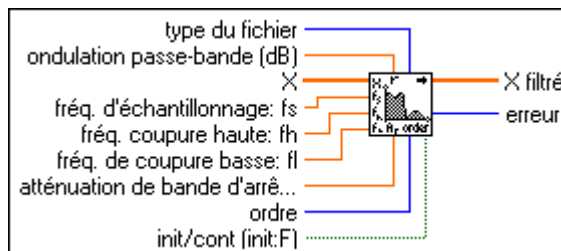
Génère l'ensemble des coefficients de filtre qui implémentent un filtre RII elliptique numérique. Vous pouvez transmettre ces coefficients au VI "Filtre RII cascadié" (IIR Cascade Filter.vi).



Le VI "Coefficients elliptiques" (Elliptic Coefficients.vi) est un sous-VI du VI "Filtre elliptique" (Elliptic Filter.vi).

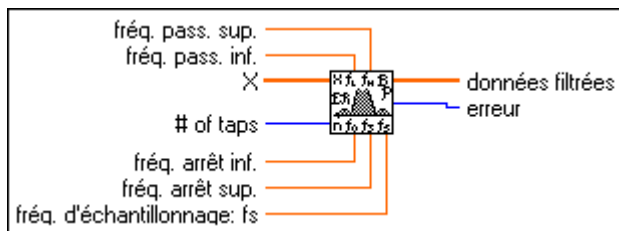
Filtre elliptique (Elliptic Filter.vi)

Génère un filtre elliptique, numérique en utilisant les paramètres **fréq. d'échantillonnage: fs**, **fréq. coupure basse: fl**, **fréq. coupure haute: fh**, **type de filtre**, **ondulation passe-bande**, **atténuation de bande d'arrêt** et **ordre** et en appelant le VI "Coefficients elliptiques" (Elliptic Coefficients.vi). Le VI "Filtre elliptique" (Elliptic Filter.vi) appelle ensuite le VI "Filtre RII" (RII Filter.vi) pour filtrer la séquence **X** en utilisant ce modèle pour obtenir une séquence **X Filtré** elliptique.



Passé-bande équi-ondulation (Equiripple BandPass.vi)

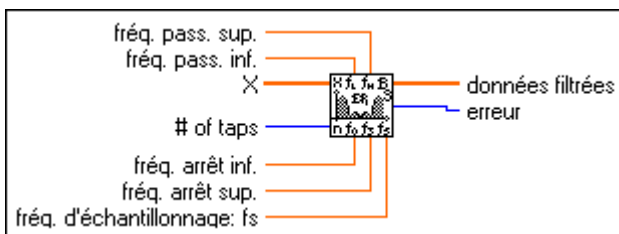
Génère un filtre RIF passé-bande pourvu de caractéristiques d'équi-ondulation en utilisant l'algorithme Parks-McClellan et les paramètres **fréq. pass. sup.**, **fréq. pass. inf.**, **ordre**, **fréq. arrêt inf.**, **fréq. arrêt sup.** et **fréq. d'échantillonnage: fs**. Le VI filtre ensuite la séquence d'entrée **X** pour obtenir la séquence **Données filtrées** en phase linéaire, filtrée en bande passante.



La première région coupe-bande du filtre est comprise entre zéro (CC) et **fréq. arrêt inf.** La région passe-bande est comprise entre **fréq. pass. inf.** et **fréq. pass. sup.** et la seconde région coupe-bande est comprise entre **fréq. arrêt sup.** et la fréquence de Nyquist.

Coupe-bande équi-ondulation (Equiripple BandStop.vi)

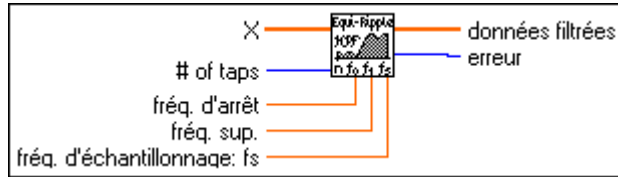
Génère un filtre RIF numérique coupe-bande pourvu de caractéristiques d'équi-ondulation en utilisant l'algorithme Parks-McClellan et les paramètres **fréq. passe sup.**, **fréq. passe inf.**, **ordre**, **fréq. arrêt inf.**, **fréq. arrêt sup.** et **fréq. d'échantillonnage: fs**. Le VI filtre ensuite la séquence d'entrée **X** pour obtenir la séquence **Données filtrées** en phase linéaire, filtrée en coupe-bande.



La première région passe-bande du filtre est comprise entre zéro (CC) et **fréq. passe inf.** La région coupe-bande est comprise entre **fréq. arrêt inf.** et **fréq. arrêt sup.** et la seconde région passe-bande est comprise entre **fréq. arrêt sup.** et la fréquence de Nyquist.

Passé-haut équi-ondulation (Equiripple HighPass.vi)

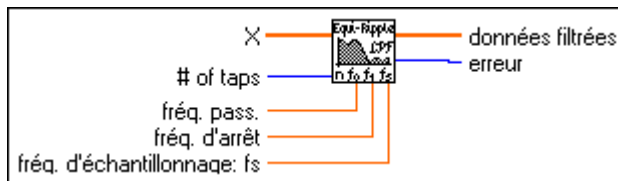
Génère un filtre RIF passé-haut pourvu de caractéristiques d'équi-ondulation en utilisant l'algorithme Parks-McClellan et les paramètres **ordre**, **fréq. d'arrêt**, **fréq. haute** et **fréq. d'échantillonnage: fs**. Le VI filtre ensuite la séquence d'entrée **X** pour obtenir la séquence **Données filtrées** en phase linéaire, filtrée en passé-haut.



La région coupe-bande du filtre est comprise entre zéro (CC) et **fréq. d'arrêt**. La bande de transition est comprise entre **fréq. d'arrêt** et **fréq. haute** et la région passe-bande est comprise entre **fréq. haute** et la fréquence de Nyquist.

Passes-bas équi-ondulation (Equiripple LowPass.vi)

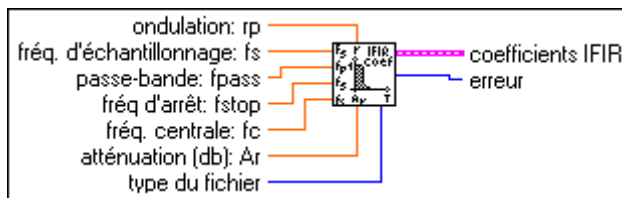
Génère un filtre RIF passe-bas pourvu de caractéristiques d'équi-ondulation en utilisant l'algorithme Parks-McClellan et les paramètres **ordre**, **fréq. pass.**, **fréq. d'arrêt** et **fréq. d'échantillonnage: fs**. Le VI filtre ensuite la séquence d'entrée **X** pour obtenir la séquence **Données filtrées** en phase linéaire, filtrée en passe-bas.



La région passe-bande du filtre est comprise entre zéro (CC) et **fréq. de pass.** La bande de transition est comprise entre **fréq. pass.** et **fréq. d'arrêt** et la région coupe-bande est comprise entre **fréq. d'arrêt** et la fréquence de Nyquist.

Coefficients bande étroite RIF (FIR Narrowband Coefficients.vi)

Génère un ensemble de coefficients de filtre qui implémentent un filtre RIF interpolé numérique. Vous pouvez transmettre ces coefficients au VI "Filtre de bande étroite RIF" (FIR Narrowband Filter.vi) pour filtrer les données.



Les figures suivantes illustrent comment les paramètres de filtre de bande étroite définissent les filtres passe-bas, passe-haut, passe-bande et coupe-bande. L'ondulation passe-bande est représentée par S_p . La réponse du filtre sur l'axe Y est représentée sur une échelle linéaire. C'est pour cette raison que l'atténuation coupe-bande A_r est représentée sous la forme d'une atténuation linéaire selon les équations suivantes :

$$A_r = -20 \log \delta_A$$

$$\delta_A = 10^{\frac{-A_r}{20}}$$

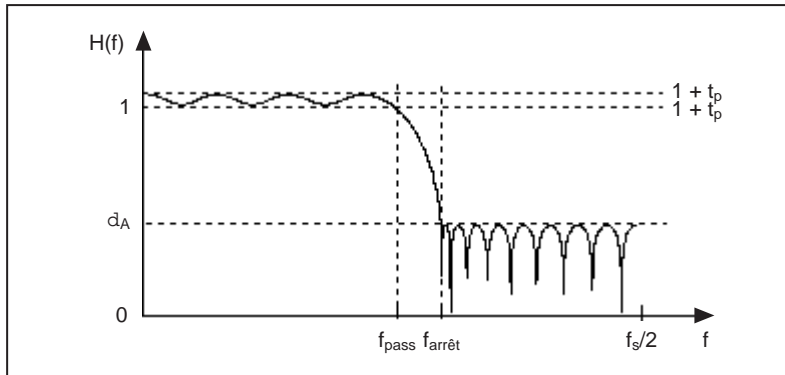


Figure 41-1. Filtre passe-bas

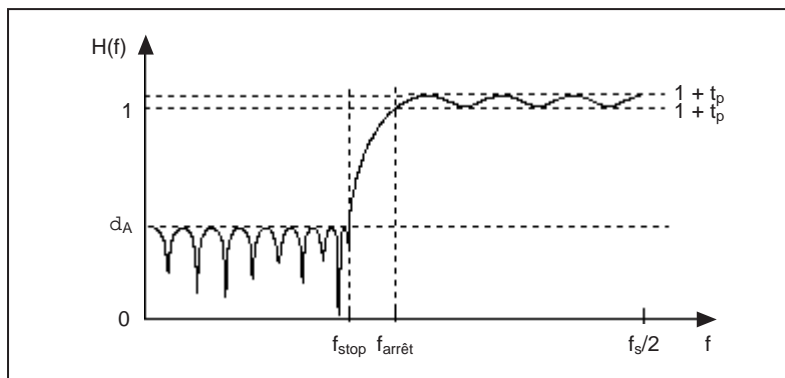


Figure 41-2. Filtre passe-haut

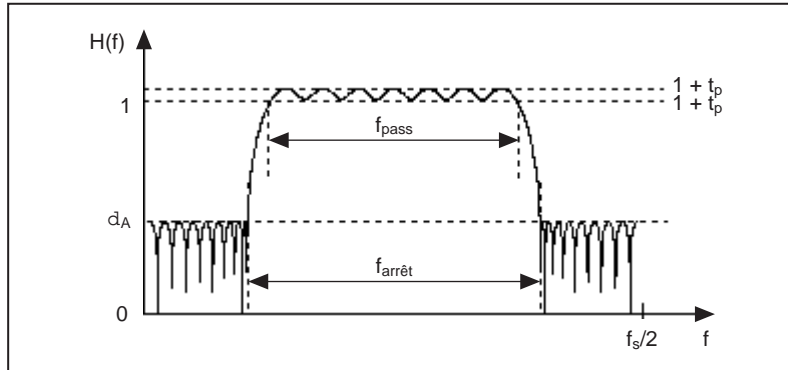


Figure 41-3. Filtre passe-bande

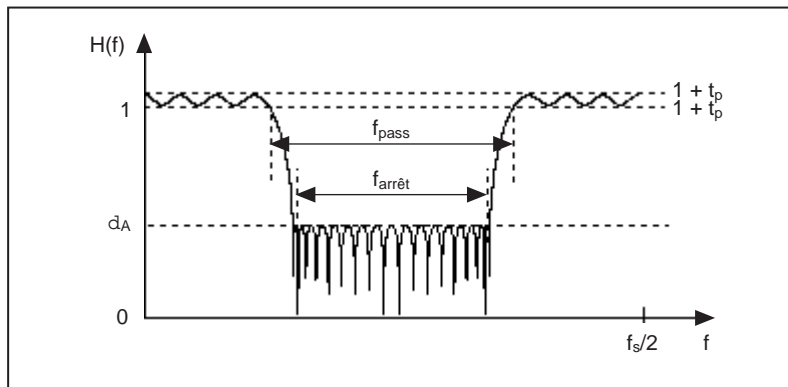
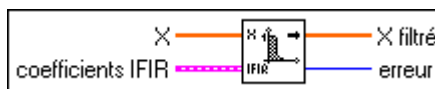


Figure 41-4. Filtre coupe-bande

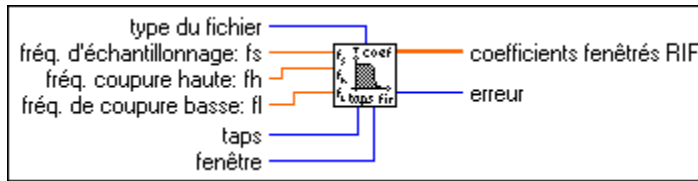
Filtre bande étroite RIF (FIR Narrowband Filter.vi)

Filtre la séquence d'entrée **X** en utilisant le filtre IFIR spécifié par les **Coefficients IFIR** définis par le VI "Coefficients de filtre bande étroite RIF" (FIR Narrowband Filter Coefficients.vi).



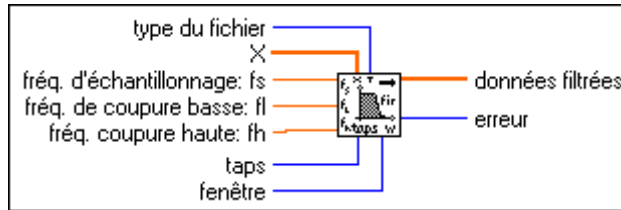
Coefficients fenêtrés RIF (FIR Windowed Coefficients.vi)

Génère l'ensemble des coefficients de filtre requis pour implémenter un filtre fenêtré RIF.



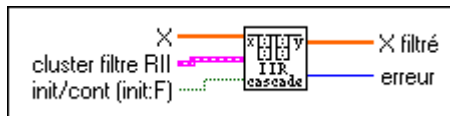
Filtre fenêtré RIF (FIR Windowed Filter.vi)

Filtre la séquence de données d'entrée, **X**, en utilisant l'ensemble des coefficients de filtre fenêtrés RIF spécifié par les paramètres **fréq. d'échantillonnage: fs**, **fréq. coupure basse: fl**, **fréq. coupure haute: fh** et **ordre**.

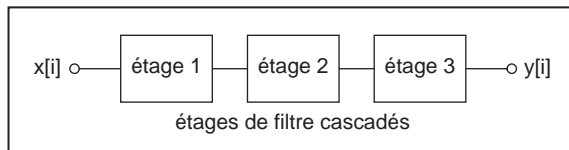


Filtre RII cascadié (IIR Cascade Filter.vi)

Filtre la séquence d'entrée **X** en utilisant la forme cascadiée du filtre RII spécifiée par le **cluster du filtre RII**.



Cette implémentation RII est dite cascadiée car il s'agit d'une cascade d'étages de filtres de second ou de quatrième ordre. La sortie d'un étage de filtre représente l'entrée de l'étage de filtre suivant pour tous les N_s étages de filtre.



Filtrage de second ordre

Chaque étage de second ordre (numéro d'étage $k = 1, 2, \dots, N_s$) comporte deux coefficients inverses (a_{1k}, a_{2k}) et trois coefficients directs (b_{0k}, b_{1k}, b_{2k}). Le nombre total de coefficients inverses est égal à $2N_s$ et le nombre total de coefficients directs est égal à $3N_s$. Le tableau de coefficients inverses et de coefficients directs comprend les coefficients d'un étage suivis des coefficients de l'étage suivant, et ainsi de suite. Par exemple, un filtre RII composé de deux étages de second ordre doit comporter un total de quatre coefficients inverses et six coefficients directs, comme suit :

coefficients inverses = $\{a_{11}, a_{21}, a_{12}, a_{22}\}$

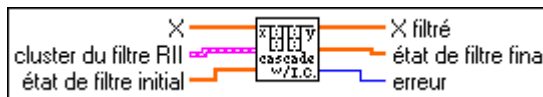
coefficients directs = $\{b_{01}, b_{11}, b_{21}, b_{02}, b_{12}, b_{22}\}$

Filtrage de quatrième ordre

Dans le cas d'une cascade d'étages de quatrième ordre, le filtrage est implémenté de la même manière qu'avec des étages de second ordre. Cependant, chaque étage doit comporter quatre coefficients inverses (a_{1k}, \dots, a_{4k}) et cinq coefficients directs (b_{0k}, \dots, b_{4k}).

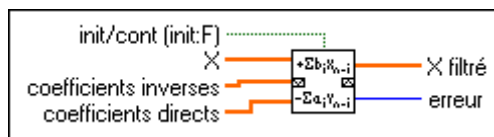
Filtre RII cascadié avec circuit intégré (IIR Cascade Filter with Integrated Circuit.vi)

Filtre la séquence d'entrée, **X**, en utilisant la forme cascadiée du filtre RII spécifié par le **cluster du filtre RII**.



Filtre RII (IIR Filter.vi)

Filtre la séquence d'entrée **X** en utilisant la forme directe du filtre RII spécifié par les **Coefficients inverses** et les **Coefficients directs**.



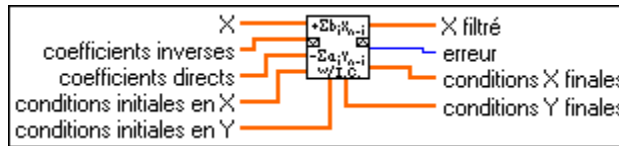
Si y représente la séquence de sortie **X Filtré**, le VI obtient les éléments y en utilisant la formule :

$$y_i = \frac{1}{a_0} \left(\sum_{j=0}^{n-1} b_j x_{i-j} - \sum_{k=1}^{m-1} a_k y_{i-k} \right)$$

où n représente le nombre de **Coefficients directs** (représenté par b_j) et m représente le nombre de **Coefficients inverses** (représenté par a_k).

Filtre RII avec circuit intégré (IIR Filter with Integrated Circuit.vi)

Filtre la séquence d'entrée **X** en utilisant la forme directe du filtre RII spécifié par les **Coefficients inverses** et les **Coefficients directs**.



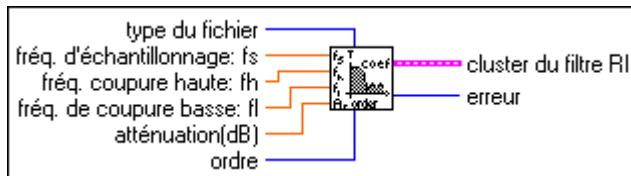
Si y représente la séquence de sortie **X Filtré**, le VI obtient les éléments y en utilisant la formule :

$$y_i = \frac{1}{a_0} \left(\sum_{j=0}^{n-1} b_j x_{i-j} - \sum_{k=1}^{m-1} a_k y_{i-k} \right)$$

où n représente le nombre de **Coefficients directs** (représenté par b_j) et m représente le nombre de **Coefficients inverses** (représenté par a_k).

Coefficients de Chebyshev inv. (Inv Chebyshev Coefficients.vi)

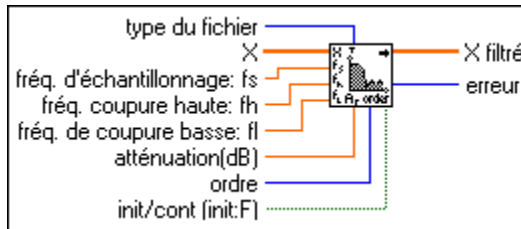
Génère l'ensemble des coefficients de filtre qui implémentent un filtre RII spécifié par le modèle de filtre de Chebyshev II. Vous pouvez transmettre ces coefficients au VI "Filtre RII cascadié" (IIR Cascade Filter.vi) pour filtrer une séquence de données.



Le VI “Coefficients de Chebyshev inv.” (Inv Chebyshev Coefficients.vi) est un sous-VI du VI “Filtre de Chebyshev inv.” (Inverse Chebyshev Filter.vi).

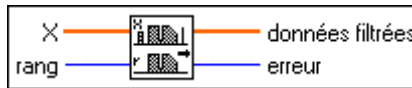
Filtre de Chebyshev inv. (Inverse Chebyshev Filter.vi)

Génère un filtre de Chebyshev II numérique utilisant les paramètres **fréq. d’échantillonnage: fs**, **fréq. coupure haute: fh**, **fréq. coupure basse: fl**, **atténuation** tels que spécifiés en décibels, **type de filtre** et **ordre du filtre** en appelant le VI “Coefficients de Chebyshev inv.” (Inv Chebyshev Coefficients.vi). Le VI “Filtre de Chebyshev inv.” (Inverse Chebyshev Filter.vi) filtre la séquence d’entrée **X** et utilise ce modèle pour obtenir une séquence **X Filtré** **Chebyshev II** en appelant le VI “Filtre RII” (IIR Filter.vi).



Filtre médian (Median Filter.vi)

Applique un filtre médian de **rang** spécifié à la séquence d’entrée **X**.



Si Y représente la séquence de sortie **Données filtrées** et si J_i représente un sous-ensemble de la séquence d’entrée **X** centrée autour du $i^{\text{ème}}$ élément de **X**

$$J_i = \{x_{i-r}, x_{i-r+1}, \dots, x_{i-1}, x_i, x_{i+1}, \dots, x_{i+r-1}, x_{i+r}\}$$

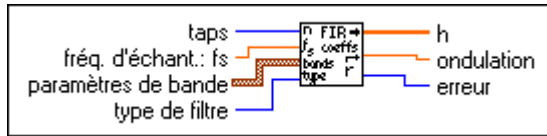
et si les éléments indexés hors de la gamme de **X** sont égaux à zéro, le VI obtient les éléments de y en utilisant

$$y_i = \text{médian}(J_i) \text{ pour } i = 0, 1, 2, \dots, n - 1$$

où n représente le nombre d’éléments dans la séquence d’entrée **X** et r représente le **rang** du filtre.

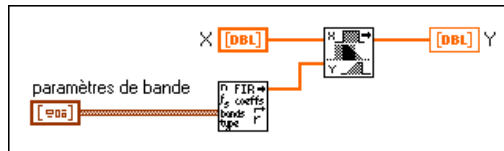
Parks-McClellan

Génère un ensemble de coefficients de filtre à phase linéaire RIF numérique et à bande multiple en utilisant les paramètres **ordre**, **fréq. d'échantillonnage: fs**, **Paramètres de bande** et **type de filtre**.



Remarque *Ce VI détermine les coefficients en utilisant des techniques itératives basées sur un critère d'erreur. Même si vous spécifiez des paramètres de filtre valides, l'algorithme peut ne pas converger.*

Le VI Parks-McClellan ne génère que des coefficients de filtre. Il n'effectue pas la fonction de filtrage. Pour filtrer une séquence **X** en utilisant l'ensemble de coefficients de filtre RIF **h**, utilisez le VI Convolution avec **X** et **h** comme séquences d'entrées.

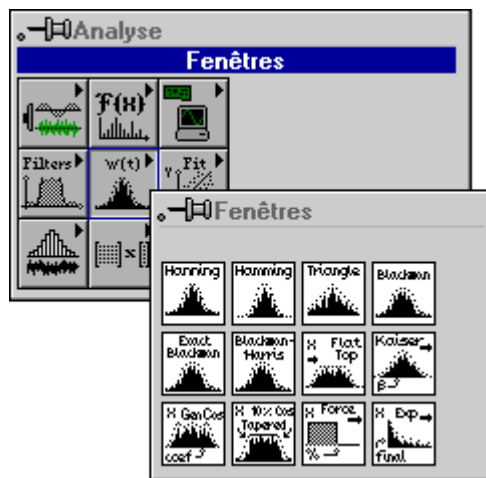


Le filtre d'équi-ondulation utilise une technique comparable pour filtrer les données.

Vis Fenêtres

Ce chapitre décrit les VIs qui implémentent les fenêtres de lissage.

Pour accéder à la palette **Fenêtres**, sélectionnez **Fonctions»Analyse»Fenêtres**. L'illustration suivante présente les options disponibles dans la palette **Fenêtres**.



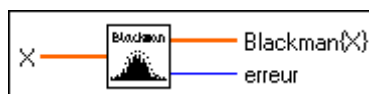
Pour des exemples sur l'utilisation des VIs Fenêtres, reportez-vous à la bibliothèque `examples\analysis\windxmpl.llb`.

Descriptions des VIs Fenêtres

Les VIs Fenêtres suivants sont disponibles.

Fenêtre de Blackman (Blackman Window.vi)

Applique une fenêtre de Blackman à la séquence d'entrée **X**.



Si y représente la séquence de sortie **Blackman{X}**, le VI obtient les éléments de y à partir de

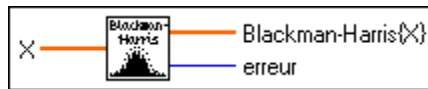
$$y_i = x_i [0,42 - 0,50 \cos(w) + 0,08 \cos(2w)] \quad \text{pour } i = 0, 1, 2, \dots, n-1$$

$$w = \frac{2\pi i}{n}$$

où n représente le nombre d'éléments de **X**.

Fenêtre de Blackman-Harris (Blackman-Harris Window.vi)

Applique une fenêtre unifiée de Blackman-Harris à la séquence d'entrée **X**.



Si y représente la séquence de sortie **Blackman-Harris{X}**, le VI obtient les éléments de y à partir de

$$y_i = x_i [0,42323 - 0,49755 \cos(w) + 0,07922 \cos(2w)]$$

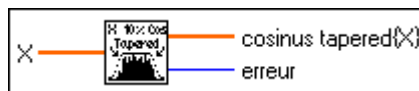
pour $i = 0, 1, 2, \dots, n-1$

$$w = \frac{2\pi i}{n}$$

où n représente le nombre d'éléments de **X**.

Fenêtre en pic de cosinus (Cosine Tapered Window.vi)

Applique une fenêtre en pic de cosinus à la séquence d'entrée **X**.



Si y représente la séquence de sortie **Pic de cosinus{X}**, le VI obtient les éléments de y à partir de

$$y_i = \begin{cases} 0,5x_i(1 - \cos w) & \text{pour } i = 0, 1, 2, \dots, m-1, \text{ et pour } i = n-m, n-m+1, \dots, n-1 \\ x_i & \text{ailleurs} \end{cases}$$

$$\text{où } w = \frac{2\pi i}{n}$$

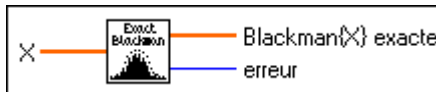
$$m = \text{rond}\left(\frac{n}{10}\right)$$

où n représente le nombre d'éléments de la séquence d'entrée \mathbf{X} .

Utiliser cette fenêtre équivaut à appliquer une fenêtre de Hanning au premier et dernier 10% de la séquence d'entrée \mathbf{X} .

Fenêtre de Blackman exacte (Exact Blackman Window.vi)

Applique une fenêtre de Blackman exacte à la séquence d'entrée \mathbf{X} .



Si y représente la séquence de sortie **Blackman exacte**{ \mathbf{X} }, le VI obtient les éléments de y à partir de

$$y_i = x_i [a_0 - a_1 \cos(w) + a_2 \cos(2w)] \text{ pour } i = 0, 1, 2, \dots, n-1$$

$$w = \frac{2\pi i}{n}$$

où n représente le nombre d'éléments de \mathbf{X} , $a_0 = 7938/18608$, $a_1 = 9240/18608$ et $a_2 = 1430/18608$.

Fenêtre exponentielle (Exponential Window.vi)

Applique une fenêtre exponentielle à la séquence d'entrée \mathbf{X} .



Si y représente la séquence de sortie **Exponentielle**{ \mathbf{X} }, le VI obtient les éléments de y à partir de

$$y_i = x_i \exp(ai) \text{ pour } i = 0, 1, 2, \dots, n-1$$

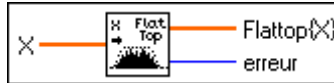
$$a = \frac{\ln(f)}{n-1}$$

où f représente la **valeur finale** et n représente le nombre d'échantillons de \mathbf{X} .

Vous pouvez utiliser ce VI pour analyser des valeurs transitoires.

Fenêtre à profil plat (Flat Top Window.vi)

Applique une fenêtre à profil plat à la séquence d'entrée \mathbf{X} .



Si y représente la séquence de sortie **Profil plat{X}**, le VI obtient les éléments de y à partir de

$$y_i = x_i [0,2810639 - 0,5208972 \cos(w) + 0,1980399 \cos(2w)]$$

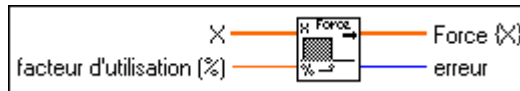
pour $i = 0, 1, 2, \dots, n-1$

$$w = \frac{2\pi i}{n}$$

où n représente le nombre d'éléments de \mathbf{X} .

Fenêtre de force (Force Window.vi)

Applique une fenêtre de force à la séquence d'entrée \mathbf{X} .



Si y représente la séquence de sortie **Force{X}**, le VI obtient les éléments de y à partir de

$$y_i = \begin{cases} x_i & \text{si } 0 \leq i \leq d \\ 0 & \text{ailleurs} \end{cases} \quad \text{pour } i = 0, 1, 2, \dots, n-1$$

$d = (0,01)(n)(\text{facteur d'utilisation (\%)}),$ où n représente le nombre d'éléments de \mathbf{X} .

Vous pouvez également utiliser ce VI pour analyser des valeurs transitoires.

Fenêtre cosinus générale (General Cosine Window.vi)

Applique une fenêtre cosinus générale à la séquence d'entrée \mathbf{X} .



Si a représente la séquence d'entrée de **Coefficients cosinus** et y représente la séquence de sortie **Cosinus générale{X}**, le VI obtient les éléments de y à partir de

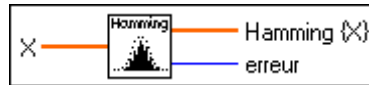
$$y_i = x_i \sum_{k=0}^{m-1} (-1)^k a_k \cos(kw) \quad \text{pour } i = 0, 1, 2, \dots, n-1$$

$$w = \frac{2\pi i}{n}$$

où n représente le nombre d'éléments de **X** et m représente le nombre de **Coefficients cosinus**.

Fenêtre de Hamming (Hamming Window.vi)

Applique une fenêtre de Hamming à la séquence d'entrée **X**.



Si y représente la séquence de sortie **Hamming{X}**, le VI obtient les éléments de y à partir de

$$y_i = x_i [0,54 - 0,46 \cos(w)] \quad \text{pour } i = 0, 1, 2, \dots, n-1$$

$$w = \frac{2\pi i}{n}$$

où n représente le nombre d'éléments de la séquence d'entrée **X**.

Fenêtre de Hanning (Hanning Window.vi)

Applique une fenêtre de Hanning à la séquence d'entrée **X**.



Si y représente la séquence de sortie **Hanning {X}**, le VI obtient les éléments de y à partir de

$$y_i = 0,5 x_i [1 - \cos(w)] \quad \text{pour } i = 0, 1, 2, \dots, n-1$$

$$w = \frac{2\pi i}{n}$$

où n représente le nombre d'éléments de \mathbf{X} .

Fenêtre de Kaiser-Bessel (Kaiser-Bessel Window.vi)

Applique une fenêtre de Kaiser-Bessel à la séquence d'entrée $\mathbf{X}(t)$.



Si y représente la séquence de sortie, **Kaiser-Bessel{X(t)}**, le VI obtient les éléments de y à partir de

$$y_i = x_i \frac{I_0(\beta \sqrt{1.0 - a^2})}{I_0(\beta)} \text{ pour } i = 0, 1, 2, \dots, n - 1$$

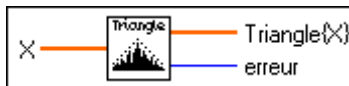
$$a = \frac{i - k}{k}$$

$$k = \frac{n - 1}{2}$$

où n représente le nombre d'éléments de $\mathbf{X}(t)$ et $I_0(\bullet)$ représente la fonction modifiée de Bessel d'ordre zéro.

Fenêtre triangulaire (Triangle Window.vi)

Applique une fenêtre triangulaire à la séquence d'entrée \mathbf{X} .



Remarque *La fenêtre de lissage triangulaire est également connue sous le nom de fenêtre de lissage de Bartlett.*

Si y représente la séquence de sortie **Triangle{X}**, le VI obtient les éléments de y à partir de

$$y_i = x_i \text{tri}(w) \text{ pour } i = 0, 1, 2, \dots, n - 1$$

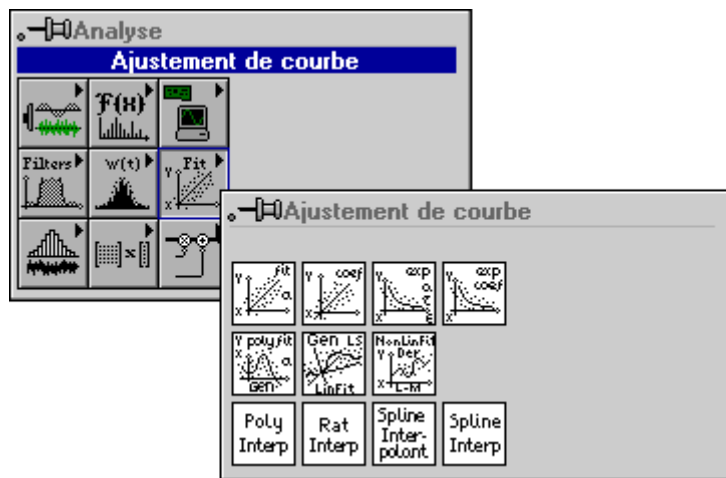
$$w = \frac{2i - n}{n}$$

où $\text{tri}(w) = 1 - |w|$ et n représente le nombre d'éléments de \mathbf{X} .

Vis d'ajustement de courbe

Ce chapitre décrit les VIs utilisés pour effectuer des opérations d'ajustement de courbe ou d'analyse de régression.

Pour accéder à la palette **Ajustement de courbe**, sélectionnez **Fonctions**» **Analyse**» **Ajustement de courbe**, comme l'indique l'illustration suivante.



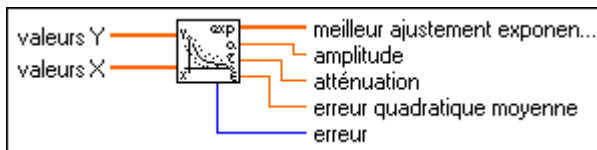
Pour des exemples sur l'utilisation des VIs de régression, reportez-vous à la bibliothèque `examples\analysis\regressn.llb`.

Descriptions des VIs d'ajustement de courbe

Les VIs d'ajustement de courbe suivants sont disponibles.

Ajustement exponentiel (Exponential Fit.vi)

Trouve les valeurs de courbe exponentielle et l'ensemble de coefficients exponentiels **amplitude** et **atténuation**, qui décrivent la courbe exponentielle représentant le mieux l'ensemble de données d'entrée.



La forme générale de l'ajustement exponentiel provient de

$$F = ae^{\tau x}$$

où F représente la séquence de sortie **Meilleur ajustement exponentiel**, X représente la séquence d'entrée **Valeurs X**, a représente l'**amplitude**, et τ représente la constante **atténuation**.

Le VI obtient l'**erreur quadratique moyenne** par le biais de la formule

$$\text{erreur quadratique moyenne} = \frac{1}{n} \sum_{i=0}^{n-1} (f_i - y_i)^2$$

où f représente la séquence de sortie **Meilleur ajustement exponentiel**, y représente la séquence d'entrée **Valeurs Y** et n représente le nombre de points de données.

Coefficients d'ajustement exponentiel (Exponential Fit Coefficients.vi)

Trouve l'ensemble de coefficients exponentiels **amplitude** et **atténuation**, qui décrivent la courbe exponentielle représentant le mieux l'ensemble de données d'entrée.



Ce VI est un sous-VI du VI "Ajustement exponentiel" (Exponential Fit.vi).

La forme générale de l'ajustement exponentiel provient de

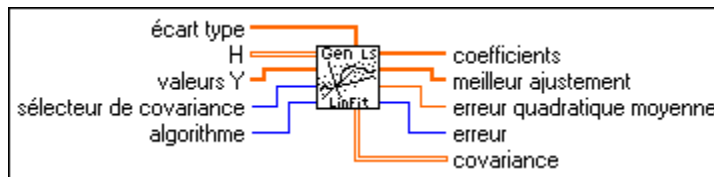
$$F = ae^{\tau x}$$

où F représente la séquence représentant les valeurs les mieux ajustées, X représente la séquence d'entrée **Valeurs X**, a représente l'**amplitude**, et τ représente la constante **atténuation**.

Ajustement linéaire général (moindre carré) (General LS Linear Fit.vi)

Trouve le plan dimensionnel k du meilleur ajustement et l'ensemble de coefficients linéaires en utilisant la méthode du moindre chi2 pour vos ensembles de données d'observation.

$\{x_{i0}, x_{i1}, \dots, x_{ik-1}, y_i\}$ où $i = 0, 1, \dots, n - 1$. n représente le nombre de vos ensembles de données d'observation.



Vous pouvez utiliser ce VI pour résoudre des problèmes de régression linéaire multiple. Vous pouvez également l'utiliser pour trouver des solutions de coefficients linéaires dans une équation à fonctions multiples.

Ajustement polynomial général (General Polynomial Fit.vi)

Trouve les valeurs de la courbe polynomiale et l'ensemble des **Coefficients d'ajustement polynomial**, qui décrivent la courbe polynomiale représentant le mieux l'ensemble de données d'entrée.



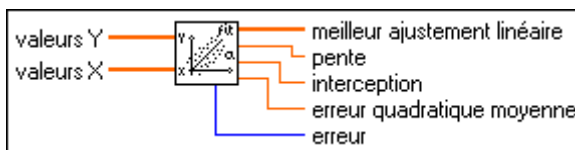
La forme générale de l'ajustement polynomial provient de

$$f_i = \sum_{j=0}^m a_j x_i^j$$

où f représente la séquence de sortie **Meilleur ajustement polynomial**, x représente la séquence d'entrée **Valeurs X**, a représente les **Coefficients d'ajustement polynomial** et m représente le paramètre **ordre polynomial**.

Ajustement linéaire (Linear Fit.vi)

Trouve les valeurs linéaires et l'ensemble de coefficients linéaires **pente** et **interception** , qui décrivent la droite représentant le mieux l'ensemble de données d'entrée.



La forme générale de l'ajustement linéaire provient de

$$F = mX + b$$

où F représente la séquence de sortie **Meilleur ajustement linéaire**, X représente la séquence d'entrée **Valeurs X**, m représente la **pente** et b représente l' **interception** .

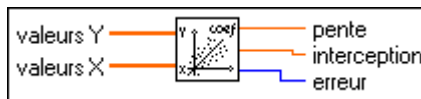
Le VI obtient la valeur **erreur quadratique moyenne** par le biais de la formule,

$$\text{erreur quadratique moyenne} = \frac{1}{n} \sum_{i=0}^{n-1} (f_i - y_i)^2$$

où F représente la séquence de sortie **Meilleur ajustement linéaire**, y représente la séquence d'entrée **Valeurs Y** et n représente le nombre de points de données.

Coefficients ajustement linéaire (Linear Fit Coefficients.vi)

Trouve l'ensemble de coefficients linéaires **pente** et **interception** , qui décrivent la droite représentant le mieux l'ensemble de données d'entrée.



Ce VI est un sous-VI du VI "Ajustement linéaire" (Linear Fit.vi).

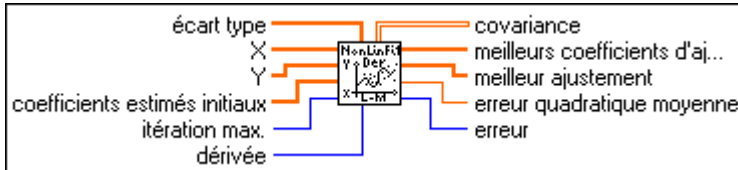
La forme générale de l'ajustement linéaire provient de

$$F = mX + b$$

où F correspond à la séquence représentant les valeurs les mieux ajustées. X représente la séquence d'entrée **Valeurs X**, m représente la **pente** et b représente l'**interception**.

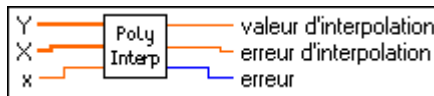
Ajustement de Lev-Mar non linéaire (Nonlinear Lev-Mar Fit.vi)

Utilise la méthode Levenberg-Marquardt pour déterminer un ensemble de coefficients non linéaires qui minimise une quantité chi-2.



Interpolation polynomiale (Polynomial Interpolation.vi)

Interpole ou extrapole la fonction f à la valeur x , suivant un ensemble de n points (x_i, y_i) , où $f(x_i) = y_i$, f représentant n'importe quelle fonction. Le VI calcule la sortie **valeur d'interpolation** $P_{n-1}(x)$, où P_{n-1} représente le polynôme unique du degré $n-1$ qui passe par les n points (x_i, y_i) .



Interpolation rationnelle (Rational Interpolation.vi)

Interpole ou extrapole f à la valeur x en utilisant une fonction rationnelle.



La fonction rationnelle

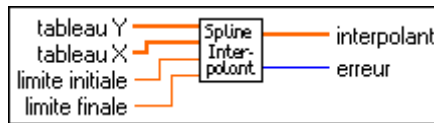
$$\frac{P(x_i)}{Q(x_i)} = \frac{p_0 + p_1x_i + \dots + p_mx_i^m}{q_0 + q_1x_i + \dots + q_vx_i^v}$$

passer par tous les points formés par le **Tableau Y** et le **Tableau X**. P et Q sont polynomiaux et la fonction rationnelle est unique, suivant un ensemble de n points (x_i, y_i) , où $f(x_i) = y_i$, f représentant n'importe quelle fonction et suivant un nombre x dans la gamme des valeurs x_i .

Ce VI calcule la sortie **valeur d'interpolation** y en utilisant $y = \frac{P(x)}{Q(x)}$. Si le nombre de points est impair, les degrés de liberté de P et Q sont égaux à $\frac{n-1}{2}$. Si le nombre de points est pair, les degrés de liberté de P sont égaux à $\frac{n}{2} - 1$ et les degrés de liberté de Q sont égaux à $\frac{n}{2}$, où n représente le nombre total de points formés par le **Tableau Y** et le **Tableau X**.

Interpolant spline (Spline Interpolant.vi)

Retourne un tableau **Interpolant** de longueur n , qui comprend les dérivées secondes de la fonction d'interpolation spline $g(x)$ aux points tabulaires x_i , où $i = 0, 1, \dots, n-1$. Les tableaux d'entrées **Tableau Y** et **Tableau X** sont de longueur n et contiennent une fonction tabulaire, $y_i = f(x_i)$, avec $x_0 < x_1 < \dots < x_{n-1}$. Les paramètres **limite initiale** et **limite finale** sont les dérivées premières de la fonction d'interpolation $g(x)$ aux points 0 et $n-1$, respectivement.



Si les paramètres **limite initiale** et **limite finale** sont égaux ou supérieurs à 10^{30} , le VI définit la condition de limite correspondante pour un spline naturel, avec zéro dérivée seconde sur cette limite.

La fonction d'interpolation $g(x)$ passe par tous les points

$$\{x_i, y_i\}, g(x_i) = y_i$$

où $i = 0, 1, \dots, n-1$.

Le VI obtient la fonction d'interpolation $g(x)$ en interpolant chaque intervalle $[x_i, x_{i+1}]$ avec une fonction polynomiale cubique $p_i(x)$ qui satisfait les conditions suivantes :

- $p_i(x_i) = y_i$
- $p_i(x_{i+1}) = y_{i+1}$
- $g(x)$ comporte des dérivées premières et secondes continues sur toute la gamme $[x_0, x_{n-1}]$:
 - $p'_i(x_i) = p'_{i+1}(x_i)$
 - $p''_i(x_i) = p''_{i+1}(x_i)$

Pour les conditions antérieures, $i = 0, 1, \dots, n-2$.

Depuis la dernière condition, $p_i''(x_i) = p_{i+1}''(x_i)$, sont dérivées les équations suivantes :

$$\frac{x_i - x_{i-1}}{6} g''(x_{i-1}) + \frac{x_{i+1} - x_{i-1}}{3} g''(x_i) + \frac{x_{i+1} - x_i}{6} g''(x_{i+1})$$

$$= \frac{y_{i+1} - y_i}{x_{i+1} - x_i} - \frac{y_i - y_{i-1}}{x_i - x_{i-1}} \quad i = 1, 2, \dots, n-2$$

Ce qui représente $n-2$ équations linéaires à n inconnues $g''(x_i)$

$i = 0, 1, \dots, n-1$. Ce VI calcule $g''(x_0)$, $g''(x_{n-1})$ à partir de la **limite initiale** et de la **limite finale** en utilisant la formule

$$g'(x) = \frac{y_{i+1} - y_i}{x_{i+1} - x_i} + \frac{3A^2 - 1}{6} (x_{i+1} - x_i) g''(x_i)$$

$$+ \frac{3B^2 - 1}{6} (x_{i+1} - x_i) g''(x_{i+1})$$

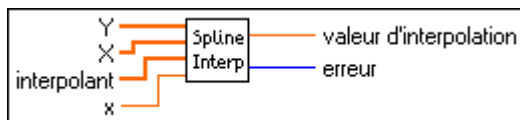
Avec

$$A = \frac{x_{i+1} - x}{x_{i+1} - x_i} \quad B = 1 - A = \frac{x - x_i}{x_{i+1} - x_i}$$

Vous pouvez dériver cette formule à partir des conditions précédentes. Ce VI utilise ensuite $g''(x_0)$, $g''(x_{n-1})$ pour déterminer tous les $g''(x_i)$, pour $i = 1, \dots, n-2$. $g''(x_i)$ représente la sortie **Interpolant**. Vous pouvez utiliser **Interpolant** comme entrée du VI "Interpolation spline" (Spline Interpolation.vi) pour interpoler y à n'importe quelle valeur de $x_0 \leq x \leq x_{n-1}$.

Interpolation spline (Spline Interpolation.vi)

Réalise une interpolation spline cubique de f à la valeur x , suivant une fonction tabulaire.



Ce VI réalise une interpolation spline cubique en utilisant une fonction tabulaire sous la forme $y = f(x)$ pour $i = 0, 1, \dots, n - 1$ en fonction des dérivées secondes **Interpolant** que le VI obtient du VI “Interpolant spline” (Spline Interpolant.vi). La valeur x doit être dans la gamme des valeurs X . Les points sont formés par les tableaux d’entrées X et Y , et n représente le nombre total de points.

Dans l’intervalle $[x_i, x_{i+1}]$, la sortie **valeur d’interpolation** y est définie par

$$y = Ay_i + By_{i+1} + Cy_i'' + Dy_{i+1}''$$

et

$$A = \frac{x_{i+1} - x}{x_{i+1} - x_i}$$

$$B = 1 - A$$

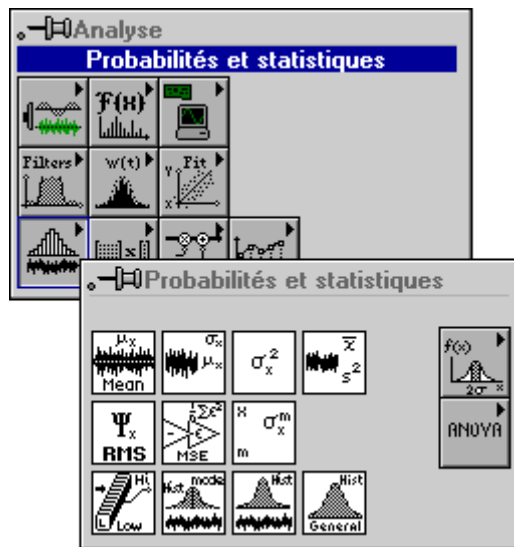
$$C = \frac{1}{6}(A^3 - A)(x_{i+1} - x_i)^2$$

$$D = \frac{1}{6}(B^3 - B)(x_{i+1} - x_i)^2$$

VIs Probabilités et statistiques

Ce chapitre décrit les VIs utilisés pour effectuer des opérations de probabilités, de statistiques descriptives, d'analyses de variance et de fonctions d'interpolation.

Pour accéder à la palette **Probabilités et statistiques**, sélectionnez **Fonctions»Analyse»Probabilités et statistiques**, comme le montre l'illustration suivante.



Pour des exemples sur l'utilisation des VIs de statistiques, reportez-vous à la bibliothèque `exemples\analyse\statxmpl.11b`.



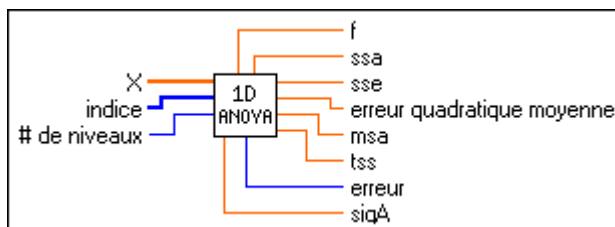
Remarque Ces VIs ne sont pas disponibles dans le kit d'Analyse de base.

Descriptions des VIs Probabilités et statistiques

Les VIs Probabilités et statistiques suivants sont disponibles.

ANOVA 1D (1D ANOVA.vi)

Prend un tableau d'observations expérimentales **X** réalisées sur plusieurs niveaux d'un facteur, avec au moins une observation par niveau, et réalise une analyse simple de variance dans le modèle à effet fixe. Lors de l'analyse simple de variance, le VI teste si le niveau de facteur produit un effet sur la sortie expérimentale.



Facteurs et niveaux

Un facteur constitue une base pour classifier des données. Par exemple, si vous comptez le nombre d'“abdominaux” qu'une personne peut faire, une des bases de classification est l'âge, pouvant comporter les niveaux suivants :

- Niveau 0 entre 6 et 10 ans
- Niveau 1 entre 11 et 15 ans
- Niveau 2 entre 16 et 20 ans

Supposez maintenant que vous effectuez une série d'observations pour noter combien d'“abdominaux” des personnes peuvent faire. Si vous échantillonnez cinq personnes au hasard, vous pouvez obtenir par exemple les résultats suivants :

Personne 1	8 ans (niveau 0)	10 “abdominaux”
Personne 2	12 ans (niveau 1)	15 “abdominaux”
Personne 3	16 ans (niveau 2)	20 “abdominaux”
Personne 4	20 ans (niveau 2)	25 “abdominaux”
Personne 5	13 ans (niveau 1)	17 “abdominaux”

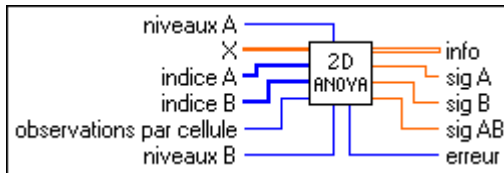
Notez que vous avez réalisé au moins une observation par niveau. Pour effectuer une analyse de variance, vous devez en effet effectuer au moins une observation par niveau.

Pour exécuter l'analyse de variance, vous devez spécifier un tableau **X** d'observations, contenant les valeurs 10, 15, 20, 25 et 17. Le tableau **Indice** spécifie le niveau (ou catégorie)

où chaque observation s'applique. Dans ce cas, **Indice** comporte les valeurs 0, 1, 2, 2 et 1. Finalement, comme il existe trois niveaux possibles, vous devez donc transmettre une valeur de 3 au paramètre **# de niveaux**.

ANOVA 2D (2D ANOVA.vi)

Prend un tableau d'observations expérimentales réalisées sur plusieurs niveaux de deux facteurs et réalise une analyse double de la variance.



Facteurs, niveaux et cellules

Un facteur constitue une base pour classifier des données. Par exemple, si vous comptez le nombre d'“abdominaux” qu'une personne peut faire, une des bases de classification est l'âge, pouvant comporter les niveaux suivants :

Niveau 0 entre 6 et 10 ans

Niveau 1 entre 11 et 15 ans

Un autre facteur possible est le poids, dont les niveaux peuvent être les suivants :

Niveau 0 moins de 50 kg

Niveau 1 entre 50 et 75 kg

Niveau 2 plus de 75 kg

Supposez maintenant que vous effectuez une série d'observations pour noter combien d'“abdominaux” des personnes peuvent faire. Si vous échantillonnez n personnes au hasard, vous pouvez obtenir par exemple les résultats suivants :

Personne 1	8 ans (niveau 0)	30 kg (niveau 0)	10 “abdominaux”
Personne 2	12 ans (niveau 1)	40 kg (niveau 0)	15 “abdominaux”
Personne 3	15 ans (niveau 1)	76 kg (niveau 2)	20 “abdominaux”
Personne 4	14 ans (niveau 1)	60 kg (niveau 1)	25 “abdominaux”
Personne 5	9 ans (niveau 0)	51 kg (niveau 1)	17 “abdominaux”
Personne 6	10 ans (niveau 0)	80 kg (niveau 2)	4 “abdominaux”

et ainsi de suite.

Si vous tracez les observations sous la forme de fonction du facteur A et du facteur B, elles sont présentées sous la forme d'une matrice où le facteur A est représenté par des rangées et le facteur B par des colonnes. Chaque cellule doit comprendre au moins une observation et chaque cellule doit comprendre le même nombre d'observations.

Pour effectuer l'analyse de variance, vous devez spécifier un tableau d'observations **X**, avec les valeurs 10, 15, 20, 25, 17 et 4. Le tableau **Indice A** spécifie le niveau (ou catégorie) du facteur A auquel s'applique chaque observation. Dans ce cas, le tableau doit comporter les valeurs 0, 1, 1, 1, 0 et 0.

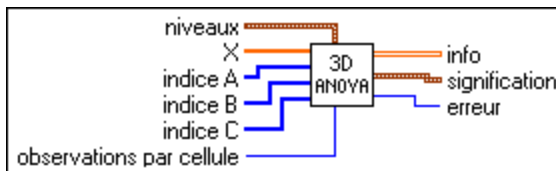
Le tableau **Indice B** spécifie le niveau (ou catégorie) du facteur B auquel s'applique chaque observation. Dans ce cas, le tableau doit comporter les valeurs 0, 0, 2, 1, 1 et 2. Finalement, comme il existe deux niveaux possibles pour le facteur A et trois niveaux possibles pour le facteur B, vous devez donc transmettre la valeur 2 au paramètre **Niveaux A** et la valeur 3 au paramètre **Niveaux B**.

Vous pouvez appliquer l'un des modèles suivants, où *L* représente le paramètre **observations par cellule** :

- Modèle 1 : effets fixes sans interaction et avec une observation par cellule (par niveaux spécifiés *x* et *y* des facteurs A et B, respectivement).
- Modèle 2 : effets fixes avec interaction et *L* > 1 observation par cellule.
- Modèle 3 : l'un des deux modèles à effets mixtes avec interaction et *L* > 1 observation par cellule.
- Modèle 4 : effets aléatoires avec interaction et *L* > 1 observation par cellule.

ANOVA 3D (3D ANOVA.vi)

Prend un tableau d'observations expérimentales réalisées sur plusieurs niveaux de trois facteurs et réalise une analyse triple de la variance. Avec toute instance de ANOVA, vous cherchez la preuve que les facteurs ou les interactions entre facteurs ont un effet significatif sur les sorties expérimentales. Ce qui différencie les modèles est la méthode utilisée.



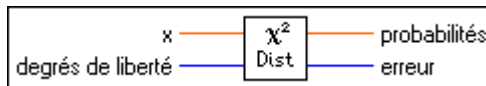
Les modèles ANOVA triples sont les suivants, où L est le nombre d'observations par cellule :

- Effets fixes avec interaction et $L > 1$ observation par cellule.
- N'importe lequel des six modèles à effets mixtes avec interaction et $L > 1$ observation par cellule.
- Effets aléatoires avec interaction et $L > 1$ observation par cellule.

Un facteur constitue une base pour classifier des données. Une cellule de données contient toutes les observations expérimentales relatives à certains niveaux des trois facteurs. Le nombre d'observations dans une cellule doit être un chiffre constant, L , qui ne varie pas entre cellules. Reportez-vous à la description des facteurs, des niveaux et des cellules dans la section sur le VI "ANOVA 2D" (2D ANOVA.vi). Souvenez-vous qu'une cellule dans ce VI "ANOVA 3D" (3D ANOVA.vi) représente l'intersection de trois facteurs au lieu de deux comme décrit dans la section du VI "ANOVA 2D" (2D ANOVA.vi).

Distribution en chi2 (Chi Square Distribution.vi)

Calcule la valeur unilatérale probabilités, p , de la variable aléatoire x à distribution χ^2 , avec des degrés de liberté spécifiés.

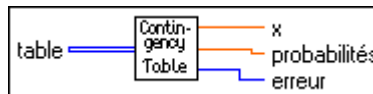


$$p = \text{Prob} \{X \leq x\}$$

où X représente χ^2 distribué sur n degrés de liberté, p représente la probabilité, n représente les degrés de liberté et x représente la valeur.

Table de contingence (Contingency Table.vi)

Classifie et inventorie des objets d'expérimentation selon deux schémas de catégorisation.



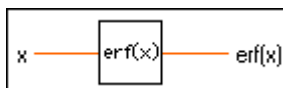
Avec le test d'homogénéité χ^2 , le VI sélectionne un échantillon aléatoire d'une certaine taille fixe dans chacune des catégories d'un schéma de catégorisation. Pour chacun des échantillons, le VI classe les objets d'expérimentation selon un second schéma et les inventorie. Le VI teste l'hypothèse pour déterminer si les distributions des populations de chaque échantillon sont identiques par rapport au second schéma de catégorisation.

Avec le test d'indépendance χ^2 , le VI ne sélectionne qu'un échantillon de la population totale. Le VI classe ensuite chaque objet et le répertorie dans deux schémas de catégorisation. Le VI teste l'hypothèse selon laquelle les schémas de catégorisation sont indépendants.

Vous devez sélectionner un niveau de signification pour chaque test. Le niveau détermine la probabilité que le VI rejette quand elle est vraie. Vous ne souhaitez pas, en général, un haut degré de probabilité. Utilisez donc un chiffre peu élevé (0,05 ou 5 % d'habitude) pour déterminer le niveau de signification. Le paramètre de sortie **probabilité** représente le niveau de signification à partir duquel l'hypothèse est rejetée. Ainsi, si la **probabilité** est inférieure au niveau de signification, vous devez rejeter l'hypothèse.

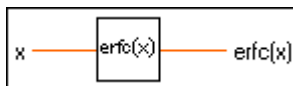
erf(x)

Evalue la fonction d'erreur à la valeur d'entrée.



erfc(x)

Evalue la fonction d'erreur complémentaire à la valeur d'entrée.

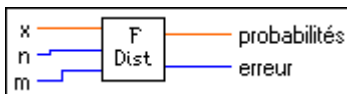


Distribution de type F (F Distribution.vi)

Calcule la valeur unilatérale **probabilité**, p , de la variable aléatoire F de distribution F , avec les degrés de libertés spécifiés n et m .

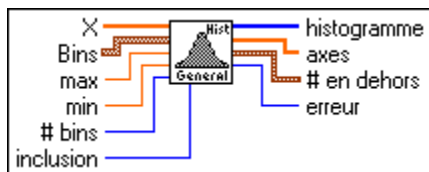
$$p = \text{Prob} \{ F_{n,m} \leq x \}$$

où F représente la distribution F , p représente la **probabilité**, n spécifie le premier degré de liberté, m spécifie le second degré de liberté et x représente la valeur.



Histogramme général (General Histogram.vi)

Détermine l'histogramme discret de la séquence d'entrée X selon les spécifications d'intervalle.



Le VI obtient l'**Histogramme** de la façon suivante. Le VI établit d'abord tous les intervalles selon les informations du tableau d'entrées **Intervalles**. Les intervalles sont :

$$\Delta_i = (\mathbf{Intervalles}[i].\text{inférieur} : \mathbf{Intervalles}[i].\text{supérieur}) \quad i = 0, 1, 2, \dots, k-1$$

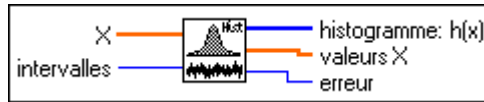
où

Intervalles[i].inférieur est la valeur **inférieur** du $i^{\text{ème}}$ cluster du tableau **Intervalles**, **Intervalles**[i].supérieur est la valeur **supérieur** du $i^{\text{ème}}$ cluster du tableau **Intervalles**, k est le nombre d'éléments dans **Intervalles**, qui est composé du nombre total d'intervalles.

Que les deux points de terminaison **Intervalles**[i].inférieur et **Intervalles**[i].supérieur de chaque intervalle fassent partie de l'intervalle Δ_i dépend de la valeur d'**inclusion d'intervalle** dans le cluster correspondant i de **Intervalles**.

Histogramme

Trouve l'histogramme discret de la séquence d'entrée **X**. L'histogramme représente un compte de la fréquence à laquelle un intervalle spécifié surgit dans la séquence d'entrée.



Si la séquence d'entrée est

$$\mathbf{X} = \{0, 1, 3, 3, 4, 4, 4, 5, 5, 8\}$$

la valeur **Histogramme: h(x)** de **X** pour huit **intervalles** est alors

$$h(X) = \{h_0, h_1, h_2, h_3, h_4, h_5, h_6, h_7\} = \{1, 1, 0, 2, 3, 2, 0, 1\}$$

Notez que l'histogramme de la séquence d'entrée **X** est une fonction de **X**.

Le VI obtient **Histogramme : h(x)** de la façon suivante. Le VI balaye la séquence d'entrée **X** pour déterminer la gamme des valeurs qu'elle contient. Le VI établit ensuite la largeur d'intervalle, Δx , selon le nombre spécifié dans **intervalles**,

$$\Delta x = \frac{\max - \min}{m}$$

où \max est la valeur maximale trouvée dans la séquence d'entrée **X**, \min est la valeur minimale trouvée dans la séquence d'entrée **X** et m représente le nombre spécifié dans **intervalles**.

Supposez que χ représente la séquence de sortie **Valeurs X**. Le VI évalue les éléments de χ en utilisant

$$\chi_i = \min + 0,5\Delta x + i\Delta x \text{ pour } i = 0, 1, 2, \dots, m-1.$$

Le VI définit le $i^{\text{ème}}$ intervalle Δ_i comme la gamme de valeurs de $\chi_i - 0,5\Delta x$ à $\chi_i + 0,5\Delta x$, non compris,

$$\Delta_i = (\chi_i - 0,5\Delta x : \chi_i + 0,5\Delta x), \text{ pour } i = 0, 1, 2, \dots, m-1$$

et définit la fonction $y_i(x)$ comme

$$y_i(x) = \begin{cases} 1 & \text{si } x \in \Delta_i \\ 0 & \text{ailleurs} \end{cases}$$

La fonction est égale à 1 si la valeur de x fait partie de l'intervalle spécifié. Sinon elle est égale à zéro. Notez que l'intervalle Δ_i est centré autour de χ_i et que sa largeur est Δ_x .

Le dernier intervalle, Δ_{m-1} , est défini comme $[\chi_{m-1} - 0,5\Delta x : \chi_{m-1} + 0,5\Delta x]$. Autrement dit, si une valeur est égale à max , elle fait partie du dernier intervalle.

Finalement, le VI évalue la séquence de l'histogramme H en utilisant

$$h_i = \sum_{j=0}^{n-1} y_i(x_j) \text{ pour } i = 0, 1, 2, \dots, m-1$$

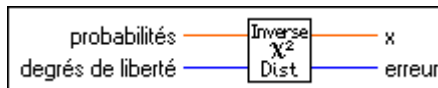
où h_i représente les éléments de la séquence de sortie **Histogramme: h(x)** et n représente le nombre d'éléments de la séquence d'entrée **X**.

Distribution en chi2 inv (Inv Chi Square Distribution.vi)

Calcule la valeur de **x** de façon à satisfaire la condition

$$p = \text{Prob} \{X \leq \mathbf{x}\}$$

pour la valeur **probabilité** p d'une variable aléatoire X de distribution χ^2 , et avec n **degrés de liberté**.

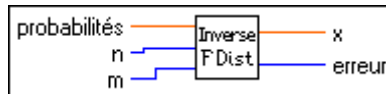


Distribution de type F inv (Inv F Distribution.vi)

Calcule la valeur de x de façon à satisfaire la condition

$$p = \{Prob_{n,m} \leq X\}$$

pour la valeur **probabilité** p d'une variable aléatoire F de distribution F, et avec n et m degrés de liberté.

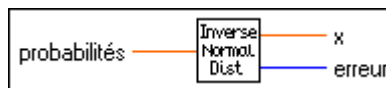


Distribution normale inv. (Inv Normal Distribution.vi)

Calcule la valeur de x de façon à satisfaire la condition

$$p = Prob \{X \leq x\}$$

pour la valeur **probabilité** p d'une variable aléatoire X de distribution normale.

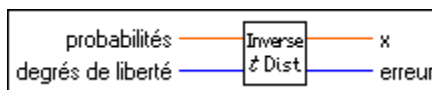


Distribution de type T inv. (Inv T Distribution.vi)

Calcule la valeur de x de façon à satisfaire la condition

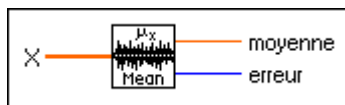
$$p = Prob \{T_n \leq x\}$$

pour la valeur **probabilité** p d'une variable aléatoire T de distribution t, et avec n **degrés de liberté**.



Moyenne (Mean.vi)

Calcule la moyenne des valeurs dans la séquence d'entrée X .



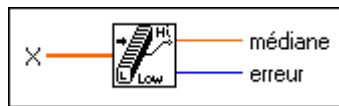
Ce VI calcule **moyenne** (μ) en utilisant la formule suivante :

$$\mu = \frac{1}{n} \sum_{i=0}^{n-1} x_i$$

où n représente le nombre d'éléments de \mathbf{X} .

Médiane (Median.vi)

Trouve la valeur médiane de la séquence d'entrée \mathbf{X} en triant les valeurs de \mathbf{X} et en sélectionnant l'élément central du tableau trié.



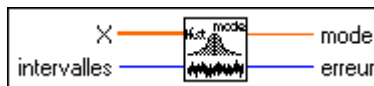
Supposez que n soit le nombre d'éléments de la séquence d'entrée \mathbf{X} et S la séquence triée de \mathbf{X} . Le VI trouve la valeur **médiane** en utilisant l'équation suivante :

$$\text{médiane} = \begin{cases} s_i & \text{si } n \text{ est impair} \\ 0,5(s_{k-1} + s_k) & \text{si } n \text{ est pair} \end{cases}$$

$$\text{où } i = \frac{n-1}{2} \text{ et } k = \frac{n}{2}$$

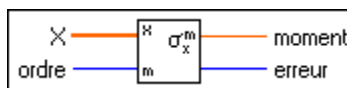
Mode

Trouve le **mode** de la séquence d'entrée \mathbf{X} .



Moment centré (Moment About Mean.vi)

Calcule le moment centré de la moyenne de la séquence d'entrée \mathbf{X} en utilisant l'**ordre** spécifié.



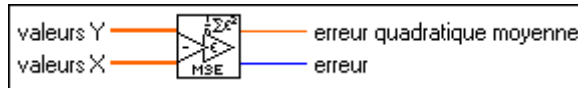
Supposez que m soit le paramètre **ordre** souhaité. Le VI calcule le **moment** d'ordre m en utilisant la formule :

$$\sigma_x^m = \frac{1}{n} \sum_{i=0}^{n-1} (x_i - \mu)^m$$

où σ_x^m est le **moment** d'ordre m , et n représente le nombre d'éléments de la séquence d'entrée \mathbf{X} .

Erreur quadratique moyenne (MSE.vi)

Calcule l'erreur quadratique moyenne (**mse**) de la séquence d'entrées **Valeurs X** et **Valeurs Y**.



Le VI utilise la formule suivante pour trouver la valeur **mse** :

$$\text{mse} = \frac{1}{n} \sum_{i=0}^{n-1} (x_i - y_i)^2$$

où n représente le nombre de points de données.

Distribution normale (Normal Distribution.vi)

Calcule la valeur unilatérale **probabilité**, p , de la variable aléatoire \mathbf{x} de distribution normale,

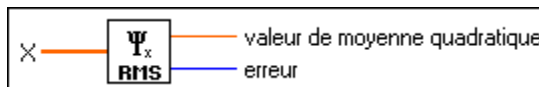
$$p = \text{Prob} \{X \leq \mathbf{x}\}$$

où X est de distribution normale standard, p représente la **probabilité** et \mathbf{x} la valeur.



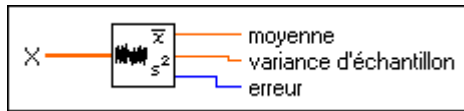
Moyenne quadratique (RMS.vi)

Calcule la moyenne quadratique (**rms**) de la séquence d'entrée \mathbf{X} .



Variance d'échantillon (Sample Variance.vi)

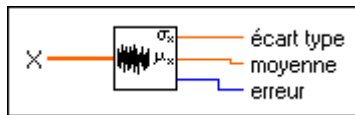
Calcule les valeurs **moyenne** et **variance d'échantillon** des valeurs de la séquence d'entrée **X**.



Remarque Si vous devez calculer l'écart-type échantillonné de **X**, utilisez la racine carrée de variance d'échantillon.

Ecart-type (Standard Deviation.vi)

Calcule la valeur moyenne et l'écart-type des valeurs de la séquence d'entrée **X**.



Ce VI calcule les valeurs **écart-type** (σ_x) et **moyenne** (μ) en utilisant la formule suivante :

$$\sigma_x = \sqrt{\frac{1}{n} \sum_{i=0}^{n-1} (x_i - \mu)^2}$$

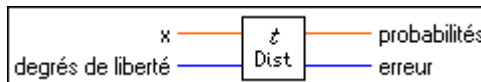
où $\mu = \frac{1}{n} \sum_{i=0}^{n-1} x_i$ et n représente le nombre d'éléments de **X**.

Distribution de type T (T Distribution.vi)

Calcule avec la valeur unilatérale **probabilité** p d'une variable aléatoire T_n de distribution t , avec les **degrés de liberté** spécifiés

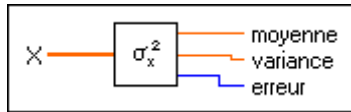
$$p = \text{Prob} \{T_n \leq x\}$$

où T est de distribution t , p représente la **probabilité**, n représente les **degrés de liberté** et x la valeur.



Variance

Calcule les valeurs de variance et de moyenne de la séquence d'entrée \mathbf{X} .



Ce VI calcule la **variance** (σ_x^2) et la **moyenne** (μ) en utilisant la formule suivante :

$$\sigma_x^2 = \frac{1}{n} \sum_{i=0}^{n-1} (x_i - \mu)^2$$

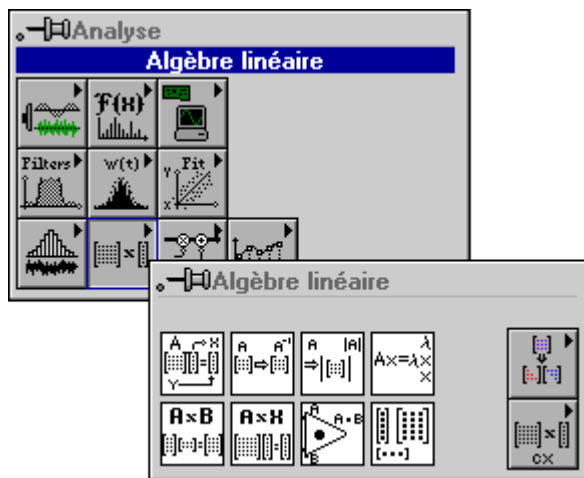
où $\mu = \frac{1}{n} \sum_{i=0}^{n-1} x_i$ et n représente le nombre d'éléments dans \mathbf{X} .

Vis d'algèbre linéaire

Ce chapitre décrit les VIs utilisés pour réaliser des opérations d'analyse et de calcul réelles et associées à des matrices, notamment les opérations suivantes :

- Manipulations de matrices de base
- Résolution d'équations linéaires et inversion de matrices
- Valeurs et vecteurs propres
- Analyses de matrices

Pour accéder à la palette **Algèbre linéaire**, sélectionnez **Fonctions»Analyse»Algèbre linéaire**, comme l'indique l'illustration suivante.



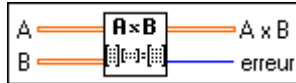
Pour des exemples sur l'utilisation des VIs d'algèbre linéaire, consultez la bibliothèque `exemples\analysis\linaxmpl.llb`.

Descriptions des VIs d'algèbre linéaire

Les VIs d'algèbre linéaire suivants sont disponibles.

A x B

Réalise la multiplication matricielle de deux matrices d'entrées.



Si A représente une matrice n -par- k et B représente une matrice k -par- m , la multiplication matricielle de A et B , $C = AB$, produit une matrice, C , dont les dimensions sont n -par- m . Supposez que A représente le tableau d'entrées à deux dimensions \mathbf{A} , B représente le tableau d'entrées à deux dimensions \mathbf{B} et C représente le tableau de sorties à deux dimensions $\mathbf{A} * \mathbf{B}$. Le VI obtient les éléments de C par le biais de la formule

$$c_{ij} = \sum_{l=0}^{k-1} a_{il}b_{lj} \quad \text{pour} \begin{cases} i = 0, 1, 2, \dots, n-1 \\ j = 0, 1, 2, \dots, m-1 \end{cases}$$

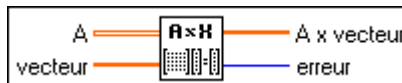
où n représente le nombre de rangées de la matrice \mathbf{A} , k représente le nombre de colonnes dans la matrice \mathbf{A} et le nombre de rangées dans la matrice \mathbf{B} , et m représente le nombre de colonnes dans la matrice \mathbf{B} .



Remarque *Le VI $A \times B$ réalise une multiplication matricielle stricte et non une multiplication à deux dimensions élément par élément. Pour effectuer une multiplication élément par élément, vous devez utiliser la fonction Multiplier de LabVIEW. En général, $AB \neq BA$.*

A x Vecteur (A x Vector.vi)

Réalise la multiplication d'une matrice d'entrée et d'un vecteur d'entrée.



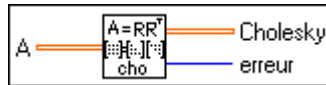
Si **A** représente une matrice n -par- k et X représente un vecteur de k éléments, la multiplication de **A** et X , $Y = AX$, produit un vecteur Y de n éléments. Supposez que Y représente la sortie **A x Vecteur**. Le VI obtient les éléments de Y par le biais de la formule :

$$y_i = \sum_{j=0}^{k-1} a_{ij}x_j, \text{ pour } i = 0, 1, 2, \dots, n-1$$

où n représente le nombre de rangées de **A**, k représente le nombre de colonnes de **A** et le nombre d'éléments de X .

Factorisation de Cholesky (Cholesky Factorization.vi)

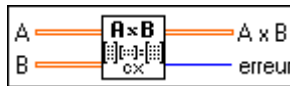
Réalise la factorisation de Cholesky d'une matrice réelle définie positive **A**.



Si la matrice réelle carrée **A** est définie positive, vous pouvez la factoriser sous la forme $A = R^T R$, où R représente une matrice supérieure triangulaire et R^T représente la transposée de R .

A x B complexe (Complex A x B.vi)

Réalise la multiplication matricielle de deux matrices d'entrée complexes.



Si A représente une matrice n -par- k et B représente une matrice k -par- m , la multiplication matricielle de A et B , $C = AB$, produit une matrice, C , dont les dimensions sont n -par- m . Supposez que A représente le tableau d'entrées à deux dimensions **A**, B représente le tableau d'entrées à deux dimensions **B** et C représente le tableau de sorties à deux dimensions **A x B**. Le VI obtient les éléments de C par le biais de la formule :

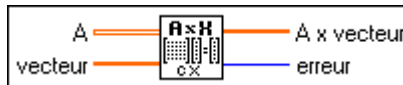
$$c_{ij} = \sum_{l=0}^{k-1} a_{il}b_{lj} \quad \text{pour } \begin{cases} i = 0, 1, 2, \dots, n-1 \\ j = 0, 1, 2, \dots, m-1 \end{cases}$$

où n représente le nombre de rangées dans la matrice **A**, k représente le nombre de colonnes dans la matrice **A** et le nombre de rangées dans la matrice **B** et m représente le nombre de colonnes dans la matrice **B**.

Remarque Le VI “A x B complexe” (Complex A x B.vi) réalise une multiplication matricielle stricte et non une multiplication à deux dimensions élément par élément. Pour effectuer une multiplication élément par élément, vous devez utiliser la fonction Multiplier de LabVIEW. En général, $AB \neq BA$.

A x vecteur complexe (Complex A x Vector.vi)

Réalise la multiplication d'une matrice d'entrée complexe et d'un vecteur d'entrée complexe.



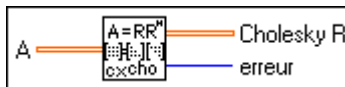
Si \mathbf{A} représente une matrice n -par- k et X représente un vecteur à k éléments, la multiplication de \mathbf{A} et X , $Y = AX$, produit un vecteur Y de n éléments. Supposez que Y représente la sortie **A x vecteur**, et X représente le vecteur d'entrée. Le VI obtient les éléments de Y par le biais de la formule :

$$y_i = \sum_{j=0}^{k-1} a_{ij}x_j, \text{ pour } i = 0, 1, 2, \dots, n-1$$

où n représente le nombre de rangées de \mathbf{A} et k représente le nombre de colonnes de \mathbf{A} et le nombre d'éléments de X .

Factorisation de Cholesky complexe (Complex Cholesky Factorization.vi)

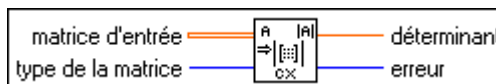
Réalise la factorisation de Cholesky d'une matrice complexe définie positive \mathbf{A} .



Si la matrice complexe carrée \mathbf{A} est définie positive, elle peut se factoriser sous la forme $A = R^H R$, où R est une matrice supérieure triangulaire et R^H représente la transposée complexe conjuguée de R .

Déterminant complexe (Complex Determinant.vi)

Trouve le **déterminant** d'une matrice complexe carrée **Matrice d'entrée**.



Supposez que A soit une matrice carrée qui représente la **Matrice d'entrée** et que L et U soient les matrices triangulaires inférieure et supérieure, respectivement, de A de telle façon que

$$A = LU$$

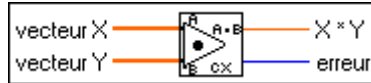
où les éléments diagonaux principaux de la matrice triangulaire inférieure L sont égaux à 1. Le VI trouve le **déterminant** de A grâce au produit des éléments diagonaux principaux de la matrice triangulaire supérieure U :

$$|A| = \prod_{i=0}^{n-1} u_{ii}$$

où $|A|$ représente le **déterminant** de A et n représente la dimension de A .

Produit scalaire complexe (Complex Dot Product.vi)

Calcule le produit scalaire des valeurs complexes **Vecteur X** et **Vecteur Y**.



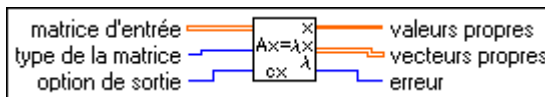
Supposez que X représente la séquence d'entrée **Vecteur X** et Y représente la séquence d'entrée **Vecteur Y**. Le VI obtient le produit scalaire X^*Y par le biais de la formule :

$$X^*Y = \sum_{i=0}^{n-1} x_i y_i$$

où n représente le nombre de points de données. Notez que la valeur de sortie X^*Y est une valeur complexe scalaire.

Vecteurs et valeurs propres complexes (Complex EigenValues & Vectors.vi)

Trouve les **Valeurs propres** et les **Vecteurs propres** droits d'une matrice d'entrée complexe carrée A .



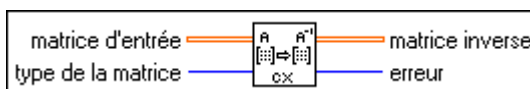
Le problème des valeurs propres consiste à déterminer les solutions non triviales de l'équation :

$$AX = \lambda X$$

où A représente une **Matrice d'entrée** n -par- n , X représente un vecteur de n éléments, et λ est un scalaire. Les valeurs n de λ qui satisfont l'équation sont les **Valeurs propres** de A et les valeurs correspondantes de X sont les **Vecteurs propres** droits de A . Une matrice hermitienne comprend toujours des valeurs propres réelles.

Matrice inverse complexe (Complex Inverse Matrix.vi)

Trouve la **Matrice inverse** d'une matrice complexe **Matrice d'entrée**.



Supposez que A soit la matrice d'entrée et I la matrice identité. Vous obtenez la **Matrice inverse** en trouvant la solution du système $AB = I$.

Si A représente une matrice non singulière, vous pouvez montrer que la solution du système précédent est unique et qu'il correspond à la matrice inverse de A

$$B = A^{-1}$$

et B représente alors la **Matrice inverse**. Une matrice non singulière est une matrice dont aucune rangée ou colonne ne contient une combinaison linéaire de toute autre rangée ou colonne, respectivement.

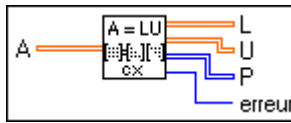


Remarque *Vous ne pouvez pas toujours déterminer d'avance si la matrice est singulière, surtout dans le cas de grands systèmes. Le VI “Matrice inverse complexe” (Complex Inverse Matrix.vi) détecte si la matrice est singulière et retourne une erreur, vous n’avez donc pas besoin de vérifier si votre système est valide avant d’utiliser ce VI.*

L’implémentation numérique de l’inversion d’une matrice est non seulement intensive sur le plan numérique, mais vu son comportement récursif, elle est également très sensible aux erreurs d’arrondi créées par le coprocesseur numérique à virgule flottante. Les calculs sont effectués avec la plus grande précision possible, mais le VI ne peut pas toujours produire des solutions au système.

Factorisation LU complexe (Complex LU Factorization.vi)

Réalise la factorisation LU d'une matrice complexe carrée **A**.



La factorisation LU sépare la matrice carrée **A** en deux matrices triangulaires dont l'une est une matrice triangulaire inférieure **L** avec des éléments diagonaux égaux à 1 et l'autre est une matrice triangulaire supérieure **U**. Ainsi,

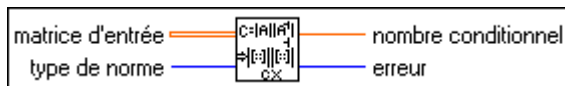
$$PA = LU$$

où **P** représente une matrice de permutation, qui consiste en la matrice d'identité dont quelques rangées sont échangées.

La factorisation est une étape clef pour inverser une matrice, calculer son déterminant et résoudre une équation linéaire.

Nombre conditionnel d'une matrice complexe (Complex Matrix Condition Number.vi)

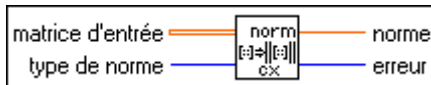
Calcule le **nombre conditionnel** d'une matrice complexe **Matrice d'entrée**.



Le **nombre conditionnel** d'une matrice mesure la sensibilité de la solution d'un système d'équations linéaires aux erreurs dans les données. Il indique le degré de précision des résultats de l'inversion de la matrice et des solutions de l'équation linéaire.

Norme d'une matrice complexe (Complex Matrix Norm.vi)

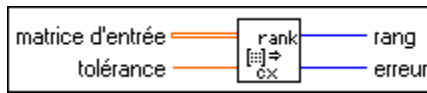
Calcule la **norme** d'une matrice complexe **Matrice d'entrée**.



La **norme** d'une matrice est un scalaire qui vous donne une indication sur l'amplitude des éléments de la matrice. Supposez que **A** représente **Matrice d'entrée**, $\|A\|_p$ représente la **norme** de **A**, où p peut être 1,2,f,∞. La valeur de p désigne le type de norme.

Rang d'une matrice complexe (Complex Matrix Rank.vi)

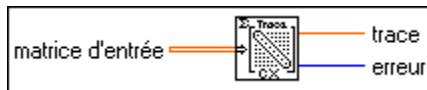
Calcule le **rang** d'une matrice complexe rectangulaire **Matrice d'entrée**.



Le **rang** représente le nombre de valeurs singulières de la **Matrice d'entrée** supérieures à la **tolérance**. Il représente donc le nombre maximal de rangées ou de colonnes indépendantes de la **Matrice d'entrée**.

Trace d'une matrice complexe (Complex Matrix Trace.vi)

Trouve la **trace** de la **Matrice d'entrée**.



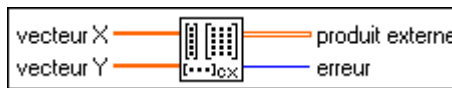
Supposez que A soit une matrice carrée qui représente la **Matrice d'entrée** et $\text{tr}(A)$ la **trace**. La **trace** de A représente la somme des éléments diagonaux principaux de A

$$\text{tr}(A) = \sum_{i=0}^{n-1} a_{ii}$$

où n représente la dimension de la **Matrice d'entrée**.

Produit externe complexe (Complex Outer Product.vi)

Calcule le produit externe des valeurs complexes **Vecteur X** et **Vecteur Y**.



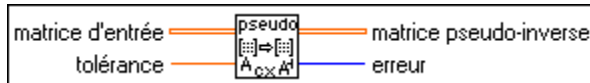
Supposez que X représente la séquence d'entrée **Vecteur X** et Y représente la séquence d'entrée **Vecteur Y**. Le VI obtient le **Produit externe** par le biais de la formule :

$$a_{ij} = x_i y_j, \text{ pour } \begin{cases} i = 0, 1, 2, \dots, n-1 \\ j = 0, 1, 2, \dots, m-1 \end{cases}$$

où A représente la séquence de sortie à deux dimensions **Produit externe**, n représente le nombre d'éléments de la séquence d'entrée **Vecteur X** et m représente le nombre d'éléments de la séquence d'entrée **Vecteur Y**.

Matrice pseudo-inverse complexe (Complex Pseudoinverse Matrix.vi)

Trouve la matrice pseudo-inverse d'une matrice complexe rectangulaire **Matrice d'entrée**.

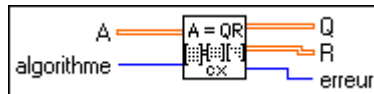


Un algorithme SVD calcule la **Matrice pseudo-inverse** A^+ et traite toute valeur singulière inférieure à la **tolérance** comme un zéro.

Si la matrice d'entrée A est carrée et non singulière, A^+ est égal à A^{-1} , cependant le VI "Matrice inverse complexe" (Complex Inverse Matrix.vi) constitue une méthode plus efficace pour calculer la valeur de A^{-1} .

Factorisation QR complexe (Complex QR Factorization.vi)

Réalise la factorisation QR d'une matrice complexe A .

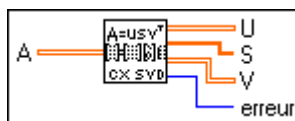


La factorisation QR est également appelée factorisation orthogonale triangulaire. Elle factorise une matrice complexe A en deux matrices ; l'une est une matrice orthogonale Q et l'autre est une matrice triangulaire supérieure R . Ainsi $A = QR$. Ce VI utilise l'algorithme de Householder.

Vous pouvez utiliser la factorisation QR pour résoudre des systèmes linéaires comprenant plus ou moins d'équations que d'inconnues.

Factorisation SVD complexe (Complex SVD Factorization.vi)

Réalise la décomposition de valeur singulière (SVD) d'une matrice A complexe m -par- n avec $m > n$.

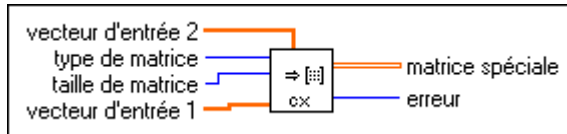


Cette décomposition produit trois matrices U , S et V . Ainsi $A = US_0V^H$, où U et V sont des matrices orthogonales, S_0 représente une matrice diagonale n -par- n contenant les éléments du

tableau **S** sur la diagonale en ordre décroissant. Les éléments diagonaux sont les valeurs singulières de **A**.

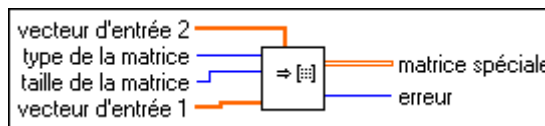
Créer une matrice complexe spéciale (Create Special Complex Matrix.vi)

Génère une matrice complexe spéciale basée sur un **type de matrice**. Les types disponibles de matrices sont Identité, Diagonal, Toeplitz et Vandermonde.



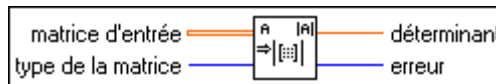
Créer une matrice spéciale (Create Special Matrix.vi)

Génère une matrice spéciale réelle basée sur un **type de matrice**. Les types disponibles de matrices sont Identité, Diagonal, Toeplitz et Vandermonde.



Déterminant (Determinant.vi)

Calcule le **déterminant** d'une matrice carrée réelle **Matrice d'entrée**.



Supposez que A soit une matrice carrée qui représente la **Matrice d'entrée**, que L et U représentent les matrices triangulaires inférieure et supérieure, respectivement, de A , de telle façon que

$$A = LU$$

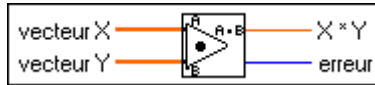
et que les éléments diagonaux principaux de la matrice triangulaire inférieure L soient égaux à 1. Le VI trouve le **déterminant** de A par le biais du produit des éléments diagonaux principaux de la matrice triangulaire supérieure U

$$|A| = \prod_{i=0}^{n-1} u_{ii}$$

où $|A|$ représente le **déterminant** de X et n représente la dimension de X .

Produit scalaire (Dot Product.vi)

Calcule le produit scalaire de **Vecteur X** et **Vecteur Y**.



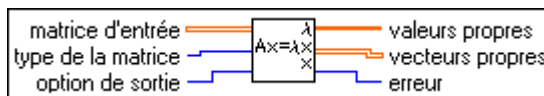
Supposez que X représente la séquence d'entrée **Vecteur X** et que Y représente la séquence d'entrée **Vecteur Y**. Le VI obtient le produit scalaire $X*Y$ par le biais de la formule :

$$X*Y = \sum_{i=0}^{n-1} x_i y_i$$

où n représente le nombre de points de données. Notez que la valeur de sortie $X*Y$ est une valeur scalaire.

Valeurs et vecteurs propres (EigenValues & Vectors.vi)

Trouve les valeurs et les vecteurs propres droits d'une **Matrice d'entrée** réelle carrée.



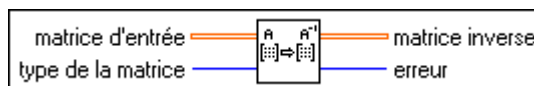
Le problème des valeurs propres consiste à déterminer les solutions non triviales à l'équation :

$$AX = \lambda X$$

où A représente une **Matrice d'entrée** n -par- n , X représente un vecteur de n éléments et λ représente un scalaire. Les n valeurs de λ qui satisfont l'équation sont les **Valeurs propres** de A et les valeurs correspondantes de X sont les **Vecteurs propres** droits de A . Une matrice réelle symétrique comporte toujours des valeurs et vecteurs propres réels.

Matrice inverse (Inverse Matrix.vi)

Trouve la **Matrice inverse** de la **Matrice d'entrée**.



Supposez que **A** soit la **Matrice d'entrée** et **I** soit la matrice d'identité. Vous obtenez la valeur de **Matrice inverse** en trouvant les solutions du système $AB = I$.

Si **A** représente une matrice non singulière, vous pouvez montrer que la solution au système précédent est unique et qu'elle correspond à la **Matrice inverse** de **A** :

$$B = A^{-1}$$

et **B** représente ainsi la **Matrice inverse**. Une matrice non singulière est une matrice dans laquelle aucune rangée ni colonne ne contient une combinaison linéaire de toute autre rangée ou colonne, respectivement.

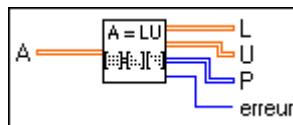


Remarque *L'implémentation numérique de l'inversion d'une matrice est non seulement intensive sur le plan numérique, mais vu son comportement récursif, elle est également très sensible aux erreurs d'arrondi créées par le coprocesseur numérique à virgule flottante. Les calculs sont effectués avec la plus grande précision possible, mais le VI ne peut pas toujours produire des solutions au système.*

Vous ne pouvez pas toujours déterminer d'avance si la matrice est singulière, surtout dans le cas de grands systèmes. Le VI "Matrice inverse complexe" (Complex Inverse Matrix.vi) détecte si la matrice est singulière et retourne une erreur, et vous n'avez donc pas besoin de vérifier si votre système est valide avant d'utiliser ce VI.

Factorisation LU (LU Factorization.vi)

Réalise la factorisation LU d'une matrice carrée réelle **A**.



La factorisation LU sépare la matrice carrée **A** en deux matrices triangulaires ; l'une est une matrice triangulaire inférieure **L** dont les éléments diagonaux sont égaux à 1 et l'autre est une matrice triangulaire supérieure **U**. Ainsi

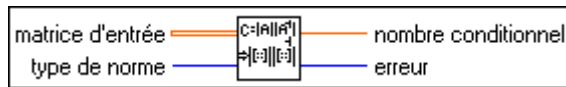
$$PA = LU$$

où **P** représente une matrice de permutation, constituée d'une matrice d'identité dont quelques rangées sont échangées.

La factorisation est une étape clef pour inverser une matrice, calculer son déterminant et résoudre une équation linéaire.

Nombre conditionnel d'une matrice (Matrix Condition Number.vi)

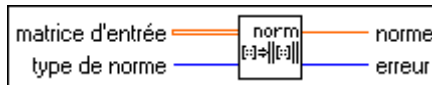
Calcule le **nombre conditionnel** d'une matrice réelle **Matrice d'entrée**.



Le **nombre conditionnel** d'une matrice mesure la sensibilité de la solution d'un système d'équations linéaires aux erreurs dans les données. Il indique le degré de précision des résultats de l'inversion de la matrice et des solutions de l'équation linéaire.

Norme d'une matrice (Matrix Norm.vi)

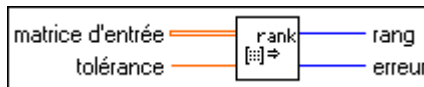
Calcule la **norme** d'une matrice réelle **Matrice d'entrée**.



La **norme** d'une matrice est un scalaire qui vous donne une indication sur l'amplitude des éléments de la matrice. Supposez que **A** représente la **Matrice d'entrée**, $\|A\|_p$ représente la **norme** de A, où p peut être 1,2,f, ∞ . La valeur p désigne le type de normes.

Rang d'une matrice (Matrix Rank.vi)

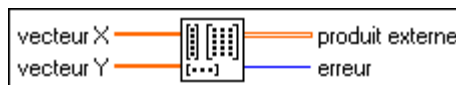
Calcule le rang d'une matrice réelle rectangulaire **Matrice d'entrée**.



Le rang de la matrice est le nombre de valeurs singulières de la **Matrice d'entrée** supérieures à la **tolérance**. Le **rang** représente le nombre maximal de rangées ou colonnes indépendantes de la **Matrice d'entrée**.

Produit externe (Outer Product.vi)

Calcule le produit externe de **Vecteur X** et **Vecteur Y**.



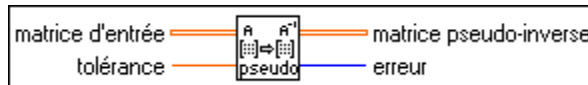
Supposez que X représente la séquence d'entrée **Vecteur X** et que Y représente la séquence d'entrée **Vecteur Y**. Le VI obtient **Produit externe** par le biais de la formule :

$$a_{ij} = x_i y_j, \text{ pour } \begin{cases} i = 0, 1, 2, \dots, n - 1 \\ j = 0, 1, 2, \dots, m - 1 \end{cases}$$

où A représente la séquence de sortie à deux dimensions **Produit externe**, n représente le nombre d'éléments de la séquence d'entrée **Vecteur X** et m représente le nombre d'éléments de la séquence d'entrée **Vecteur Y**.

Matrice pseudo-inverse (PseudInverse Matrix.vi)

Détermine la **matrice pseudo-inverse** d'une matrice réelle rectangulaire **Matrice d'entrée**.

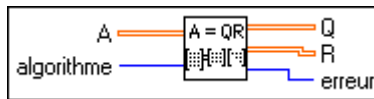


La **matrice pseudo-inverse** A^+ est calculée en utilisant l'algorithme SVD et n'importe quelle valeur singulière inférieure à **tolérance**, réglés sur zéro.

Si la matrice d'entrée A est carrée et non singulière, A^+ est égal à A^{-1} ; cependant le VI "Matrice inverse" (Inverse Matrix.vi) constitue une méthode plus efficace pour calculer la valeur de A^{-1} .

Factorisation QR (QR Factorization.vi)

Réalise la factorisation QR d'une matrice réelle A .

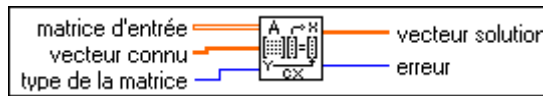


La factorisation QR est également appelée factorisation orthogonale triangulaire. Elle factorise une matrice réelle A en deux matrices. L'une est une matrice orthogonale Q et l'autre est une matrice triangulaire supérieure R , telles que $A = QR$. Ce VI fournit trois méthodes de factorisation : Householder, Givens et Givens rapide.

Vous pouvez utiliser la factorisation QR pour résoudre des systèmes linéaires avec plus d'équations que d'inconnues.

Résolution d'équations linéaires complexes (Solve Complex Linear Equations.vi)

Résout un système linéaire complexe $AX = Y$.



Supposez que A représente la **Matrice d'entrée** m -par- n , Y représente l'ensemble de m éléments du **Vecteur connu** et X représente l'ensemble de n éléments du **Vecteur de solution** qui fournit la solution du système

$$AX = Y$$

Quand $m > n$, le système comprend plus d'équations que d'inconnues et constitue donc un système surdéterminé. Etant donné que la solution qui satisfait $AX = Y$ peut ne pas exister, le VI détermine la solution de moindre carré X , qui minimise $\|AX - Y\|$.

Quand $m < n$, le système comprend plus d'inconnues que d'équations et constitue donc un système sous-déterminé. Il peut disposer d'un nombre infini de solutions qui satisfont $AX = Y$. Le VI sélectionne alors une de ces solutions.

Si $m = n$, et si A est une matrice non singulière (aucune rangée ou colonne ne contient une combinaison linéaire de toute autre rangée ou colonne, respectivement) vous pouvez alors résoudre le système en décomposant la **Matrice d'entrée** A en ses matrices triangulaires inférieure et supérieure, L et U , de façon à ce que

$$AX = LZ = Y$$

et


$$Z = UX$$

puissent être une autre représentation du système d'origine. Notez que Z est également un vecteur de n élément.

Les systèmes triangulaires sont faciles à résoudre avec des techniques récursives. Ainsi, quand vous obtenez les matrices L et U dans A , vous pouvez déterminer Z dans le système $LZ = Y$ et X dans le système $UX = Z$.

Quand $m \neq n$, A peut être décomposée en une matrice orthogonale Q et en une matrice triangulaire supérieure R , telles que $A = QR$ et le système linéaire peut être représenté par $QRX = Y$. Vous pouvez résoudre alors $RX = Q^H Y$.

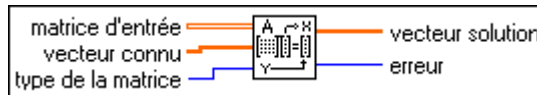
Vous pouvez facilement résoudre ce système triangulaire pour obtenir X en utilisant des techniques récursives.

 **Remarque** Vous ne pouvez pas toujours déterminer d'avance si la matrice est singulière, surtout dans le cas de grands systèmes. Le VI “*Matrice inverse complexe*” (*Complex Inverse Matrix.vi*) détecte si la matrice est singulière et retourne une erreur, vous n'avez donc pas besoin de vérifier si votre système est valide avant d'utiliser ce VI.

L'implémentation numérique de l'inversion d'une matrice est intensive sur le plan numérique, mais vu son comportement récursif, elle est également très sensible aux erreurs d'arrondi créées par le coprocesseur numérique à virgule flottante. Les calculs sont effectués avec la plus grande précision possible, mais le VI ne peut pas toujours offrir des solutions au système.

Résolution d'équations linéaires (Solve Linear Equations.vi)

Résout un système linéaire réel $AX = Y$.



Supposez que A soit une matrice m -par- n qui représente la **Matrice d'entrée**, Y soit l'ensemble de coefficients m du **Vecteur connu** et X l'ensemble d'éléments n du **Vecteur solution** qui résout le système

$$AX = Y$$

Quand $m > n$, le système comprend plus d'équations que d'inconnues et constitue donc un système surdéterminé. La solution qui satisfait $AX = Y$ peut ne pas exister, et le VI trouve donc la solution de moindre carré X , qui minimise $\|AX - Y\|$.

Quand $m < n$, le système comprend plus d'inconnues que d'équations et constitue donc un système sous-déterminé. Il peut comprendre un nombre infini de solutions qui satisfont $AX = Y$. Le VI trouve une de ces solutions.

Dans le cas où $m = n$, si A représente une matrice non singulière (aucune rangée ou colonne ne contient une combinaison linéaire de toute autre rangée ou colonne, respectivement), vous pouvez alors résoudre le système de X en décomposant la **Matrice d'entrée** A en ses matrices triangulaires inférieure et supérieure, L et U , de façon à ce que

$$AX = LZ = Y$$

et


$$Z = UX$$

puissent être une autre représentation du système d'origine. Notez que Z est aussi un vecteur de n éléments.

Les systèmes triangulaires sont faciles à résoudre avec des techniques récursives. Ainsi, quand vous obtenez les matrices L et U de A , vous pouvez trouver Z dans le système $LZ = Y$ et X dans le système $UX = Z$.

Dans le cas où $m \neq n$, A peut être décomposée en une matrice orthogonale Q et en une matrice triangulaire supérieure R , telles que $A = QR$ et le système linéaire peut être représenté par $QRX = Y$. Vous pouvez résoudre alors $RX = Q^T Y$.

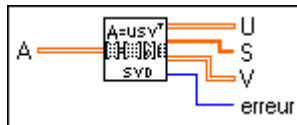
Vous pouvez facilement résoudre ce système triangulaire pour obtenir x en utilisant des techniques récursives.

 **Remarque** *Vous ne pouvez pas toujours déterminer d'avance si la matrice est singulière, surtout dans le cas de grands systèmes. Le VI "Matrice inverse complexe" (Complex Inverse Matrix.vi) détecte si la matrice est singulière et retourne une erreur, et vous n'avez donc pas besoin de vérifier si votre système est valide avant d'utiliser ce VI.*

L'implémentation numérique de l'inversion d'une matrice est non seulement intensive sur le plan numérique, mais vu son comportement récursif, elle est également très sensible aux erreurs d'arrondi créées par le coprocesseur numérique à virgule flottante. Les calculs sont effectués avec la plus grande précision possible, mais le VI ne peut pas toujours produire des solutions au système.

Factorisation SVD (SVD Factorization.vi)

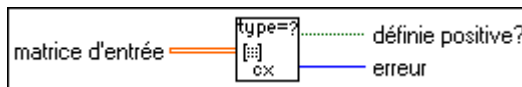
Réalise la décomposition en valeur singulière (SVD) d'une matrice A donnée m -par- n réelle, avec $m > n$.



La décomposition produit trois matrices U , S_0 et V telles que $A = US_0V^T$, où U et V^T sont des matrices orthogonales, S_0 représente une matrice diagonale n -par- n contenant les éléments diagonaux du tableau S en ordre décroissant.

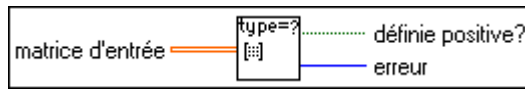
Matrice complexe définie positive-test (Test Complex Positive Definite.vi)

Teste si la **Matrice d'entrée** est une matrice définie positive.



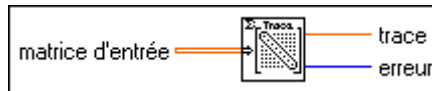
Matrice définie positive-test (Test Positive Definite.vi)

Teste si la **Matrice d'entrée** est une matrice définie positive.



Trace

Détermine la **trace** de la **Matrice d'entrée**.



Supposez que A soit une matrice carrée qui représente la **Matrice d'entrée** et $\text{tr}(A)$ soit la **trace**. La **trace** de A est la somme des éléments diagonaux principaux de A

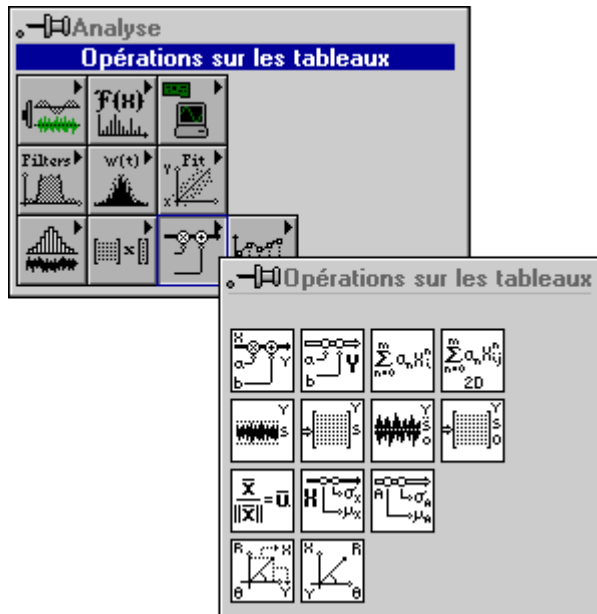
$$\text{tr}(A) = \sum_{i=0}^{n-1} a_{ii}$$

où n représente la dimension de la **Matrice d'entrée**.

Vis d'opérations sur tableaux

Ce chapitre décrit les VIs utilisés pour réaliser des opérations communes sur des tableaux numériques à une et à deux dimensions.

L'illustration suivante présente la palette **Opérations sur tableaux**, à laquelle vous accédez en sélectionnant **Fonctions»Analyse»Opérations sur tableaux**.

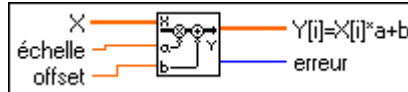


Descriptions des VIs d'opérations sur tableaux

Les VIs d'opérations sur tableaux suivants sont disponibles.

Evaluation linéaire 1D (1D Linear Evaluation.vi)

Réalise une évaluation linéaire du tableau d'entrée **X**.



Le tableau de sortie $Y[i] = X[i]*a + b$ est produit par

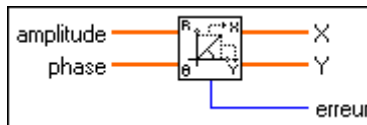
$$Y = aX + b$$

où a représente la constante multiplicative **échelle** et b représente la constante additive **offset**.

Polaire vers cartésien 1D (1D Polar To Rectangular.vi)

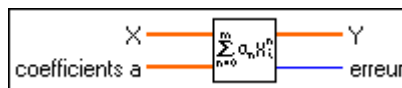
Convertit deux tableaux de coordonnées polaires en deux tableaux de coordonnées cartésiennes, selon les formules suivantes :

$$\begin{aligned} x &= \text{Magnitude} \cos(\text{Phase}) \\ y &= \text{Magnitude} \sin(\text{Phase}). \end{aligned}$$



Evaluation polynomiale 1D (1D Polynomial Evaluation.vi)

Réalise une évaluation polynomiale de **X** en utilisant les **Coefficients: a**.



Le tableau de sortie **Y** est produit par

$$Y = \sum_{n=0}^m a_n X^n$$

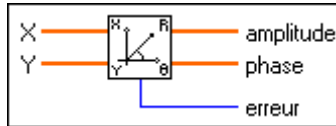
où m indique l'ordre polynomial.

Cartésien vers polaire 1D (1D Rectangular To Polar.vi)

Convertit deux tableaux de coordonnées cartésiennes en deux tableaux de coordonnées polaires, selon les formules suivantes :

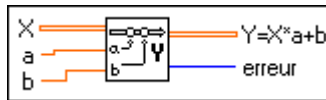
$$\text{amplitude} = \sqrt{x^2 + y^2}$$

$$\text{phase} = \tan^{-1} \left(\frac{y}{x} \right)$$



Evaluation linéaire 2D (2D Linear Evaluation.vi)

Réalise une évaluation linéaire du tableau d'entrée à deux dimensions **X**.



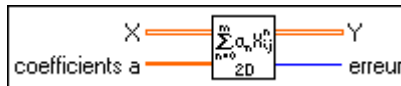
Le tableau d'entrée à deux dimensions **Y = X*a + b** est produit par

$$Y = Xa + b$$

où *a* indique la constante multiplicative et *b* indique la constante additive.

Evaluation polynomiale 2D (2D Polynomial Evaluation.vi)

Réalise une évaluation polynomiale du tableau d'entrée à deux dimensions **X** en utilisant les Coefficients **a**.



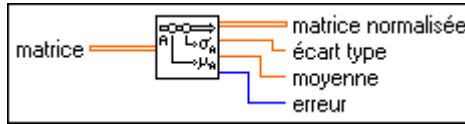
Le tableau d'entrée à deux dimensions **Y** est produit par

$$Y = \sum_{n=0}^m a_n X^n$$

où *m* indique l'ordre polynomial.

Normaliser une matrice (Normalize Matrix.vi)

Normalise la **Matrice** d'entrée à deux dimensions en utilisant son profil statistique (μ , σ), où μ représente la **moyenne** et σ représente l'**écart-type**, pour obtenir la **Matrice normalisée** dont le profil statistique est égal à (0,1).



Le VI obtient la **Matrice normalisée** en utilisant

$$B = \frac{A - \mu}{\sigma}$$

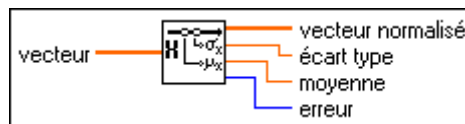
$$\mu = \frac{\sum_{i=0}^{n-1} \sum_{j=0}^{m-1} a_{ij}}{n \cdot m}$$

$$\sigma = \sqrt{\frac{\sum_{i=0}^{n-1} \sum_{j=0}^{m-1} (a_{ij} - \mu)^2}{n \cdot m}}$$

où B représente la séquence de sortie à deux dimensions **Matrice normalisée**, A représente la séquence d'entrée à deux dimensions **Matrice** composée de n rangées et m colonnes et a_{ij} représente l'élément de A dans la $i^{\text{ème}}$ rangée et la $j^{\text{ème}}$ colonne.

Normaliser un vecteur (Normalize Vector.vi)

Normalise le **Vecteur** d'entrée en utilisant son profil statistique (μ , σ), où μ représente la **moyenne** et σ représente l'**écart-type**, pour obtenir un **Vecteur normalisé** dont le profil statistique est égal à (0,1).



Le VI obtient le **Vecteur normalisé** en utilisant

$$Y = \frac{X - \mu}{\sigma}$$

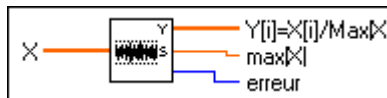
$$\mu = \frac{\sum_{i=0}^{n-1} x_i}{n}$$

$$\sigma = \sqrt{\frac{\sum_{i=0}^{n-1} (x_i - \mu)^2}{n}}$$

où Y représente la séquence de sortie **Vecteur normalisé** et X représente la séquence d'entrée **Vecteur** de longueur n et x_i représente le $i^{\text{ème}}$ élément de X .

Echelle rapide 1D (Quick Scale 1D.vi)

Détermine la valeur absolue maximum du tableau d'entrée \mathbf{X} et met ensuite \mathbf{X} à l'échelle en utilisant cette valeur.



Le tableau de sortie $\mathbf{Y}[i] = \mathbf{X}[i]/\mathbf{Max}|\mathbf{X}|$ est produit par

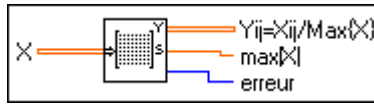
$$Y = \frac{X}{s}$$

où s représente la valeur absolue maximum de \mathbf{X} .

Vous pouvez utiliser ce VI pour normaliser des séquences à l'intérieur de la gamme $[-1,1]$. Ce VI est très utile si la séquence a une moyenne égale à zéro.

Echelle rapide 2D (Quick Scale 2D.vi)

Détermine la valeur absolue maximum du tableau d'entrée **X** et met ensuite **X** à l'échelle en utilisant cette valeur.



Le tableau de sortie $Y_{ij} = X_{ij}/\text{Max}\{X\}$ est produit par

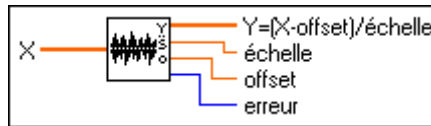
$$Y = \frac{X}{s}$$

où s indique la valeur absolue maximum de **X**.

Vous pouvez utiliser ce VI pour normaliser des séquences à l'intérieur de la gamme $[-1,1]$. Ce VI est très utile si la matrice a une moyenne égale à zéro.

Echelle 1D (Scale 1D.vi)

Détermine l'échelle et l'offset et met la matrice **X** à l'échelle en utilisant ces valeurs.



Le tableau de sortie **Y** est produit par

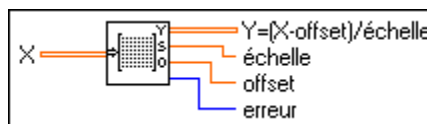
$$Y = \frac{X - \text{offset}}{\text{échelle}}$$

échelle = $0,5(\text{max} - \text{min})$ et **offset** = $\text{min} + \text{échelle}$, où *max* indique la valeur maximum de **X** et *min* indique la valeur minimum de **X**.

Vous pouvez utiliser ce VI pour normaliser n'importe quelle séquence numérique avec l'assurance que la gamme de la séquence de sortie sera $[-1,1]$.

Echelle 2D (Scale 2D.vi)

Détermine l'échelle et l'offset et met ensuite **X** à l'échelle en utilisant ces valeurs.



Le tableau de sortie à deux dimensions $\mathbf{Y} = (\mathbf{X} - \text{offset})/\text{échelle}$ est produit par

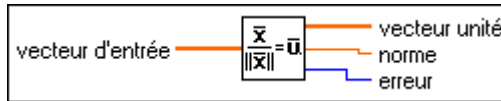
$$Y = \frac{X - \text{offset}}{\text{échelle}}$$

échelle = $0,5(\text{max} - \text{min})$ et **offset** = $\text{min} + 0,5 \text{ échelle}$, où *max* indique la valeur maximum de \mathbf{X} et *min* indique la valeur minimum de \mathbf{X} .

Vous pouvez utiliser ce VI pour normaliser n'importe quelle séquence numérique avec l'assurance que la gamme de la séquence de sortie sera $[-1,1]$.

Vecteur unité (Unit Vector.vi)

Détermine la **norme** du **Vecteur d'entrée** et obtient son **Vecteur unité** correspondant en normalisant le **Vecteur d'entrée** avec sa **norme**.



Supposez que X représente l'entrée **Vecteur d'entrée** ; **norme** est produite par

$$\|X\| = \sqrt{x_0^2 + x_1^2 + \dots + x_{n-1}^2}$$

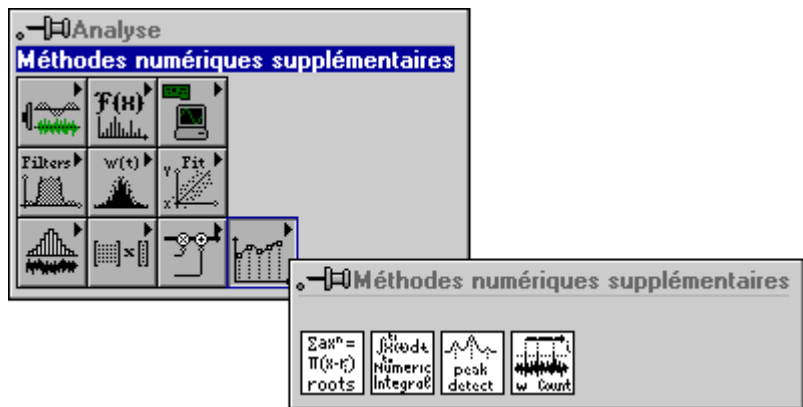
où $\|X\|$ représente la **norme** et le VI calcule **Vecteur unité**, U , en utilisant

$$U = \frac{X}{\|X\|}$$

Vis de méthodes numériques supplémentaires

Ce chapitre décrit les VIs implémentant des méthodes numériques pour réaliser des opérations de détection de crêtes, d'intégration numérique et de recherche.

L'illustration suivante présente la palette **Méthodes numériques supplémentaires**, que vous pouvez ouvrir en sélectionnant **Fonctions»Analyse»Méthodes numériques supplémentaires**.

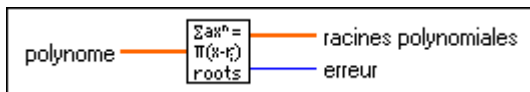


Descriptions des VIs de méthodes numériques supplémentaires

Les VIs de méthodes numériques supplémentaires suivants sont disponibles.

Racines polynomiales complexes (Complex Polynomial Roots.vi)

Trouve les racines complexes d'un polynôme complexe.

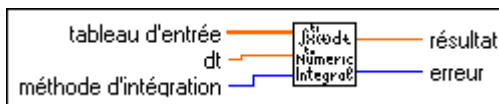


Ce VI utilise une méthode de Newton modifiée pour déterminer les racines complexes n (certaines peuvent être réelles, avec une partie imaginaire égale à zéro) du polynôme général complexe :

$$a_0 + a_1x + a_2x^2 + \dots + a_{n-1}x^{n-1} + a_nx^n$$

Intégration numérique (Numeric Integration.vi)

Réalise une intégration numérique sur le tableau d'entrée de données en utilisant l'une des quatre méthodes communes d'intégration numérique.



Remarque

Si le nombre de points fournis pour une méthode choisie ne contient pas de nombre intégral de sommes partielles, la méthode est alors appliquée à tous les points possibles. Pour les points restants, la méthode suivante d'ordre inférieur possible est utilisée. Par exemple, si la méthode de Bode est sélectionnée, le tableau suivant montre ce que ce VI calcule pour différents nombres de points :

Nombre de points	Evaluations partielles effectuées
224	56 Bode
225	56 Bode, 1 Trapézoïdal
226	56 Bode, 1 Simpson
227	56 Bode, 1 Simpson 3/8
228	57 Bode

Ainsi, si 227 points sont disponibles et que la méthode de Bode est sélectionnée, le VI produit le résultat en réalisant 56 évaluations partielles selon la méthode de Bode et une évaluation selon la méthode de Simpson 3/8.

Chacune des méthodes dépend de l'intervalle d'échantillonnage (**dt**) et calcule l'intégrale en utilisant des applications successives d'une formule de base pour exécuter des évaluations partielles, qui dépendent d'un certain nombre de points adjacents. Le nombre de points utilisés dans chaque évaluation partielle représente l'ordre de la méthode. Le résultat est la somme de ces évaluations partielles successives.

$$\text{résultat} = \int_{t_0}^{t_1} f(t) dt \approx \sum_j \text{sommes partielles}$$

où *j* représente une gamme qui dépend du nombre de points et de la méthode d'intégration.

Les formules de base pour le calcul de la somme partielle de chaque règle en ordre croissant des méthodes sont :

Trapézoïdal : $(x[i] + x[i+1]) * dt, k = 1$

Simpson: $(x[2i] + 4x[2i+1] + x[2i+2]) * dt/3, k = 2$

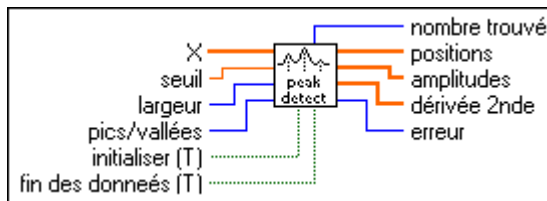
Simpson 3/8 : $(3x[3i] + 9x[3i+1] + 9x[3i+2] + 3x[3i+3]) * dt/8, k = 3$

Bode : $(14x[4i] + 64x[4i+1] + 24x[4i+2] + 64x[4i+3] + 14x[4i+4]) * dt/45, k = 4$
 pour $i = 0, k, 2k, 3k, 4k, \dots$, Partie intégrale de $[(N-1)/k]$

où *N* représente le nombre de points de données, *k* représente un entier qui dépend de la méthode et *x* représente le tableau d'entrée.

Détection de pics (Peak Detector.vi)

Trouve l'emplacement, l'amplitude et la dérivée seconde de crêtes ou de creux dans le tableau d'entrée.



L'ensemble de données peut être transmis au VI sous forme de tableau unique ou de blocs de données consécutifs.

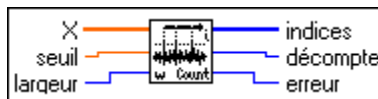
Ce VI se base sur un algorithme qui ajuste le polynôme quadratique à des groupes séquentiels de points de données. Le nombre de points de données utilisé dans cet ajustement est spécifié par **largeur**.

Pour chaque crête ou creux, l'ajustement quadratique est testé par rapport au niveau **seuil** : les crêtes à hauteurs inférieures à **seuil** ou les creux à profondeurs supérieures à **seuil** sont ignorés. Les **crêtes/creux** sont détectés seulement après le traitement de **largeur**/2 points de données au-delà des emplacements **crêtes/creux**. Ce délai n'affecte que le traitement en temps réel.

Ce VI doit être notifié lorsque les premier et dernier blocs lui sont transmis, pour que le VI puisse s'initialiser et libérer ensuite les données propres à l'algorithme de détection de crêtes.

Détection de pics avec seuillage (Threshold Peak Detector.vi)

Analyse la séquence d'entrée **X** pour détecter des crêtes valides, **compte** le nombre de crêtes trouvées et déterminent leurs **indices**, et repère les points dépassant le paramètre **seuil** dans une crête valide. Une crête est valide lorsque les éléments de **X** dépassent le paramètre **seuil** et reviennent ensuite sur une valeur inférieure ou égale à **seuil** et que le nombre d'éléments supérieurs à **seuil** est au moins égal à **largeur**.



Partie V

VIs et fonctions de communication

La Partie V, *VIs et fonctions de communication*, décrit la façon dont LabVIEW gère les communications sur réseau et d'interapplication et présente les VIs et les fonctions de communication. Cette partie contient les chapitres suivants :

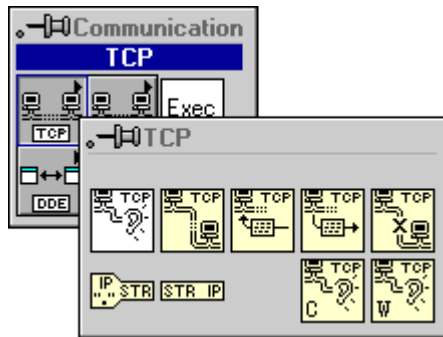
- Le chapitre 48, *VIs TCP*, décrit le Internet Protocol (protocole Internet) (IP), le Transmission Control Protocol (protocole de contrôle des transmissions) (TCP), les adresses Internet et les VIs TCP de LabVIEW. Reportez-vous au chapitre 21, *VIs TCP et UDP du Manuel de l'utilisateur LabVIEW* pour avoir une vue d'ensemble de TCP/IP ainsi que des exemples d'applications client/serveur TCP.
- Le chapitre 49, *VIs UDP*, décrit un ensemble de VIs que vous pouvez utiliser avec le User Datagram Protocol (UDP), un protocole dans la suite TCP/IP permettant de communiquer à travers un réseau unique ou un ensemble de réseaux interconnectés.
- Le chapitre 50, *VIs DDE*, décrit les VIs LabVIEW pour le Dynamic Data Exchange (DDE) (Echange dynamique de données) sous Windows 3.1, Windows 95 et Windows NT. Ces VIs exécutent des fonctions DDE permettant le partage de données avec d'autres applications qui acceptent les connexions DDE.
- Le chapitre 51, *Fonctions d'Automation ActiveX*, décrit les fonctions qui supportent l'Automation ActiveX. Ces fonctions permettent à d'autres fonctions ActiveX activées telles que Microsoft Excel, de requérir des propriétés et des méthodes auprès de LabVIEW et de VIs individuels.

- Le chapitre 52, *Vis AppleEvent*, décrit les VIs LabVIEW pour AppleEvents, une forme de communication interapplication (IAC), entre des applications Macintosh.
- Le chapitre 53, *Vis de communication entre programmes*, décrit les VIs LabVIEW utilisés pour la communication entre programmes (PPC), une forme de communication interapplication Apple de niveau bas (IAC) par laquelle des applications Macintosh envoient et reçoivent des blocs de données.

VI s TCP

Ce chapitre décrit le protocole Internet (Internet Protocol : IP), le protocole de contrôle de la transmission (Transmission Control Protocol : TCP) et l'adresse Internet. Il décrit également les VI s TCP de LabVIEW. Consultez le chapitre 21, *TCP et UDP*, du *Manuel de l'utilisateur LabVIEW* pour obtenir une vue d'ensemble de TCP/IP et des exemples d'applications TCP client/serveur.

L'illustration suivante présente la palette **TCP**, que vous pouvez ouvrir en sélectionnant **Fonctions»Communication»TCP**.



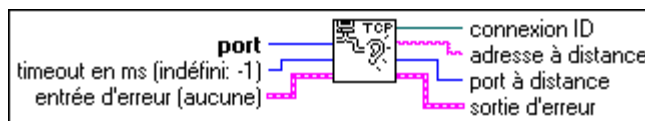
Pour des exemples sur l'utilisation des VI s TCP, reportez-vous à la bibliothèque `examples\comm\tcpex.llb`.

Description du VI TCP

Le VI TCP suivant est disponible.

TCP Listen

Crée un auditeur et attend une connexion TCP acceptée au port spécifié.



Lorsqu'une "écoute" sur un port donné débute, vous ne pouvez pas utiliser un autre VI "TCP Listen" sur le même port. Par exemple, supposez qu'un VI soit doté de deux VIs "TCP Listen" dans son diagramme. Si vous commencez une écoute sur le port 2222 avec le premier VI "TCP Listen", vous ne pourrez pas effectuer d'écoute sur le port 2222 avec le second VI "TCP Listen".

Fonctions TCP/IP

Les VIs suivants sont maintenant des fonctions dans LabVIEW 5.0 :

- IP en chaîne de caractères
- Chaîne de caractères en IP
- TCP Open Connection
- TCP Create Listener
- TCP Wait on Listener
- TCP Write
- TCP Read
- TCP Close Connection

Le VI "TCP Listen" est toujours un VI dans LabVIEW 5.0 car sa fonctionnalité est doublée par les fonctions "TCP Create Listener" et "TCP Wait on Listener".

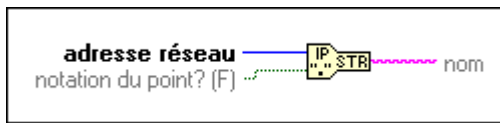
Les fonctions "TCP Write", "TCP Read" et "TCP Wait on Listener" incorporent une fonctionnalité nouvelle. Le paramètre **entrée de données** de "TCP Write" accepte maintenant des tableaux d'octets. "TCP Read" dispose d'une nouvelle entrée, **mode**, qui modifie son fonctionnement. Les quatre modes sont Standard, Bufférisé, CRLF (retour chariot/retour à la ligne) et Immédiat. "TCP Wait on Listener" dispose d'une nouvelle entrée, **résoudre l'adresse à distance**, qui indique s'il faut résoudre l'adresse à distance ou la laisser en notation scalaire.

"Standard" a le même comportement que dans les versions précédentes de LabVIEW. "Bufférisé" correspond à une lecture tout-ou-rien. Si vous n'avez pas reçu les octets demandés à la fin du timeout, aucun octet n'est retourné. Les octets non retournés sont sauvegardés pour d'autres essais de lecture. CRLF correspond à une lecture jusqu'à ce que le retour chariot et un retour à la ligne soient repérés dans le flux d'entrée. Vous devez toujours spécifier une taille maximale de lecture. Si le CRLF ne se trouve pas dans la taille indiquée, aucune donnée n'est retournée. Si la limite de timeout est atteinte et qu'aucun CRLF n'a été trouvé, aucune donnée n'est retournée. "Immédiat" exige de terminer immédiatement la lecture dès que des octets sont reçus.

Les fonctions TCP/IP suivantes sont disponibles.

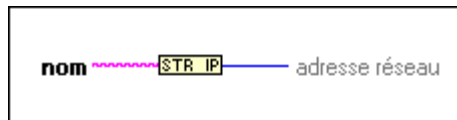
IP en chaîne de caractères

Convertit une adresse de réseau IP en chaîne de caractères.



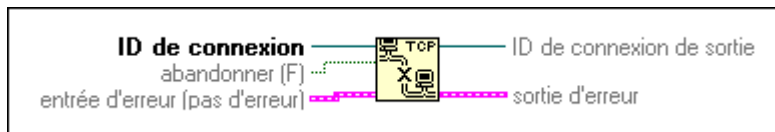
Chaîne de caractères en IP

Convertit une chaîne de caractères en une adresse de réseau IP.



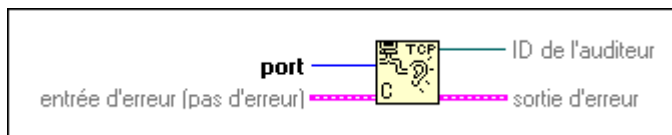
TCP Close Connection

Ferme la connexion associée avec **connexion ID**.



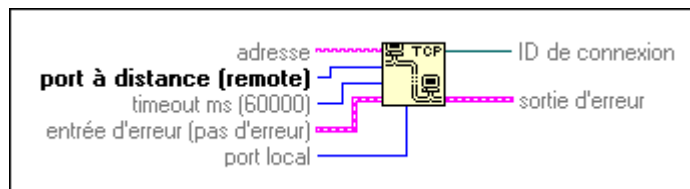
TCP Create Listener

Crée un auditeur pour une connexion TCP.



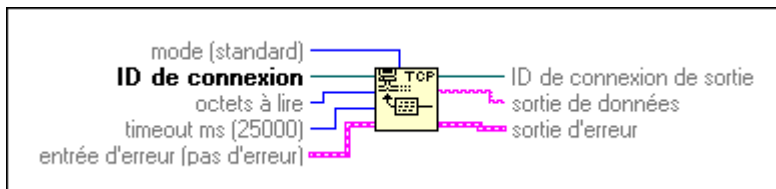
TCP Open Connection

Essaie d'ouvrir une connexion TCP avec l'adresse et le port spécifiés.



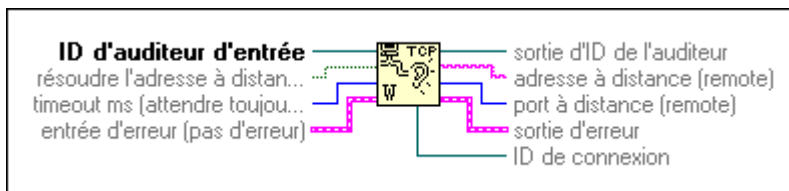
TCP Read

Reçoit jusqu'à **octets à lire** depuis la connexion TCP spécifiée et retourne les résultats dans **sortie de données**.



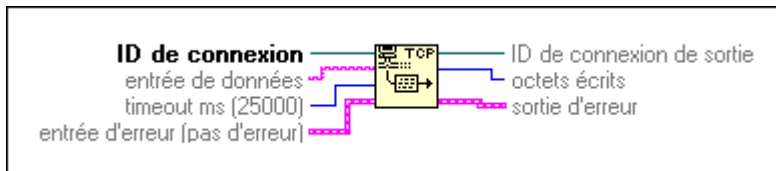
TCP Wait on Listener

Attend une connexion TCP acceptée au port spécifié.



TCP Write

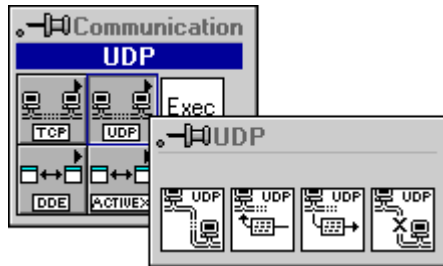
Ecrit la chaîne de caractères **entrée de données** à la connexion TCP spécifiée.



Vis UDP

Ce chapitre décrit un ensemble de VIs que vous pouvez utiliser avec le protocole User Datagram Protocol (UDP), qui fait partie de la série TCP/IP pour communiquer sur un réseau unique ou sur un ensemble de réseaux interconnectés.

L'illustration suivante présente la palette **UDP**, que vous pouvez ouvrir en sélectionnant **Fonctions»Communication»UDP**.

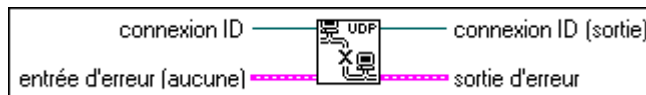


Descriptions des VIs UDP

Les VIs UDP suivants sont disponibles.

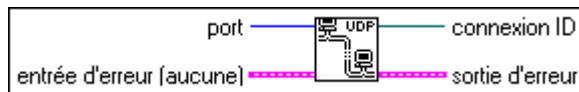
UDP Close

Ferme la connexion UDP spécifiée par **connexion ID**.



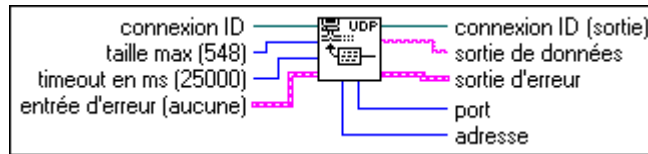
UDP Open

Essaie d'ouvrir une connexion UDP sur le **port** donné. Le paramètre **connexion ID** est un "symbole opaque" utilisé dans toutes les opérations ultérieures associées à la connexion.



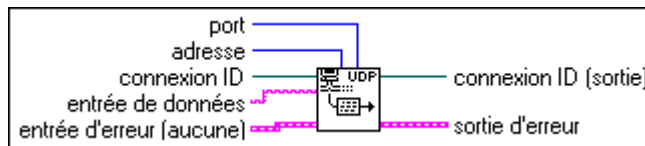
UDP Read

Retourne un datagramme (reçu sur la connexion UDP spécifiée par **connexion ID**) dans la chaîne de caractères **sortie de données**.



UDP Write

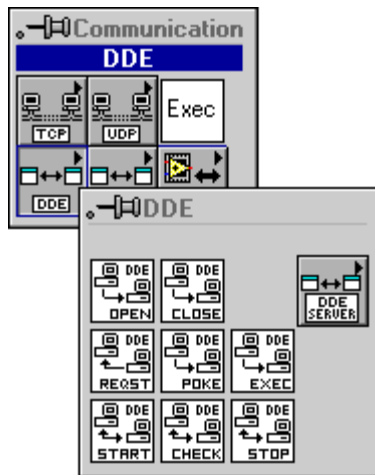
Ecrit la chaîne de caractères **entrée de données** à la connexion à distance UDP spécifiée par **adresse** et **port**.



Vis DDE

Ce chapitre décrit les VIs de LabVIEW utilisés pour l'échange dynamique de données (Dynamic Data Exchange-DDE) sous Windows 3.1, Windows 95 et Windows NT. Ces VIs exécutent des fonctions DDE pour partager des données avec d'autres applications qui acceptent des connexions DDE.

L'illustration suivante présente la palette **DDE**, que vous pouvez ouvrir en sélectionnant **Fonctions»Communication»DDE**.



La palette **DDE** comprend la sous-palette **Serveur DDE**.

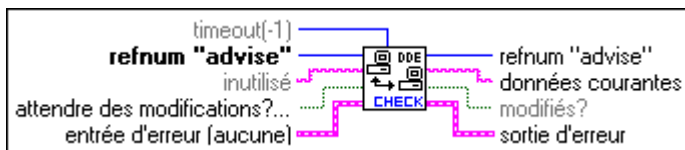
Pour des exemples sur l'utilisation des VIs DDE, consultez la bibliothèque `examples\comm\DDEexamp.llb`.

Descriptions des Vis Client DDE

Les VIs Client DDE suivants sont disponibles.

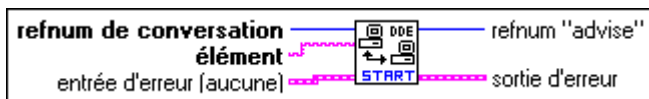
DDE Advise Check

Vérifie une valeur de type “advise” antérieurement définie par le VI DDE Advise Start.



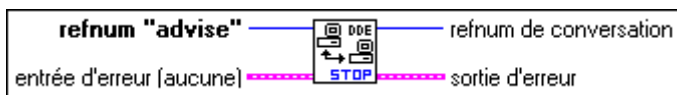
DDE Advise Start

Initie une liaison de type “advise”.



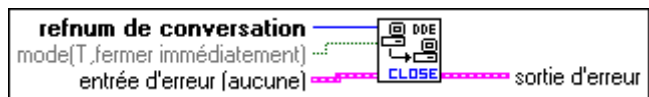
DDE Advise Stop

Annule une liaison de type “advise”, antérieurement définie par le VI DDE Advise Start.



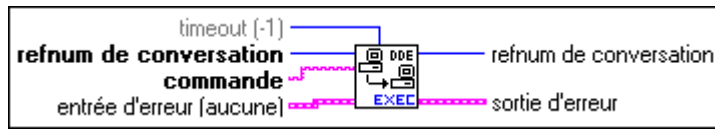
DDE Close Conversation

Termine une conversation DDE.



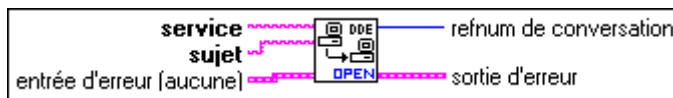
DDE Execute

Demande au serveur DDE d'exécuter la valeur **commande**.



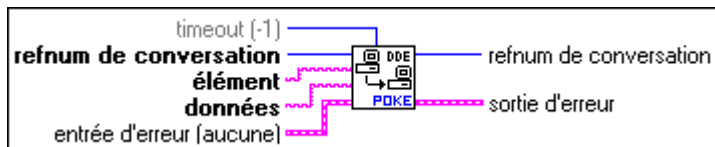
DDE Open Conversation

Définit une connexion entre LabVIEW et une autre application. Vous devez appeler ce VI avant d'utiliser tout autre VI DDE (excepté les VIs Serveur).



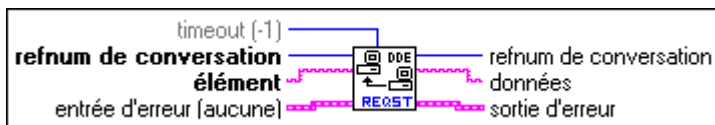
DDE Poke

Demande au serveur DDE de placer la valeur **données** dans l'**élément** spécifié.



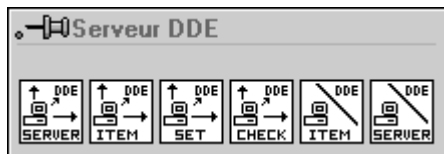
DDE Request

Initie un échange de messages DDE pour obtenir la valeur courante de l'**élément**.



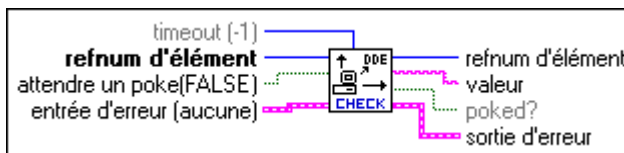
Descriptions des Vis Serveur DDE

Vous pouvez accéder aux fonctions Serveur DDE en sélectionnant **Fonctions» Communication» DDE» Serveur DDE**.



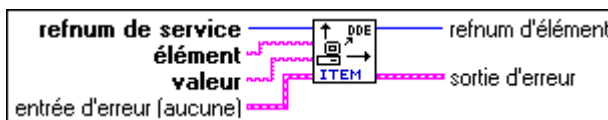
DDE Srv Check Item

Règle la valeur d'un élément DDE antérieurement défini.



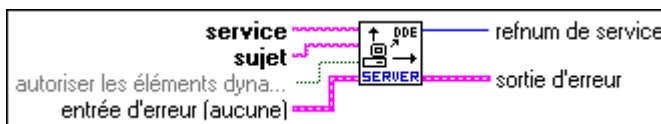
DDE Srv Register Item

Définit un élément DDE pour le service spécifié par **refnum de service**.



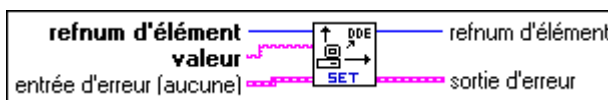
DDE Srv Register Service

Définit un service DDE auquel les clients peuvent se connecter.



DDE Srv Set Item

Règle la valeur d'un élément DDE antérieurement défini.

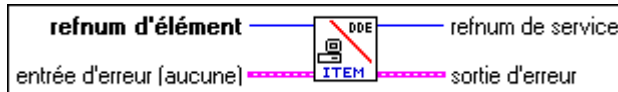


DDE Srv Unregister Item

Supprime l'élément spécifié de son service.

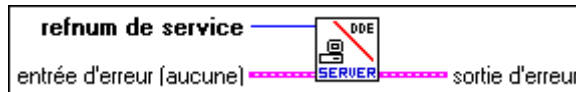


Remarque *Les clients DDE ne peuvent plus accéder à l'élément une fois que ce VI termine son exécution.*



DDE Srv Unregister Service

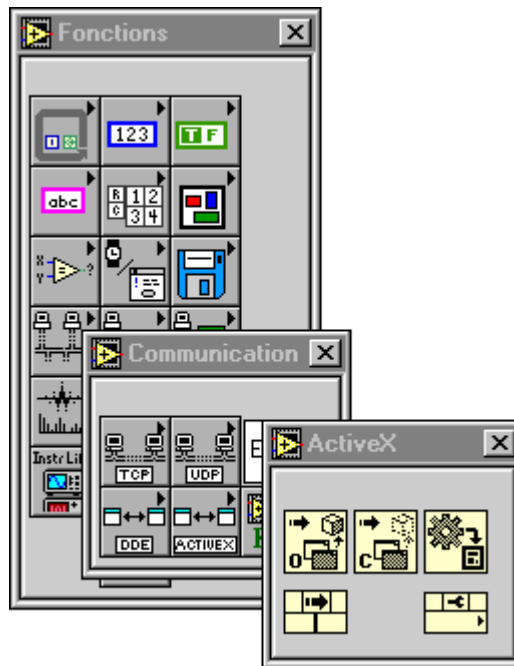
Supprime le service spécifié. Les clients DDE ne peuvent plus se connecter à ce service et toutes les conversations en cours sont interrompues.



Fonctions d'Automation ActiveX

Ce chapitre décrit les fonctions de support d'Automation ActiveX. Ces fonctions permettent aux autres applications activées par ActiveX, comme Microsoft Excel, de requérir des propriétés et des méthodes de la part de LabVIEW et de VIs individuels.

Vous pouvez accéder aux fonctions d'Automation ActiveX en sélectionnant **Fonctions»Communication»ActiveX/OLE**.



La palette **ActiveX/OLE** comprend les fonctions suivantes :

- Automation Open
- Automation Close
- Nœud de méthode
- Nœud de propriété

Elle comprend également la fonction “En données G”. Pour plus d’informations sur cette fonction, reportez-vous à la section *Fonctions de conversion des données* figurant plus loin dans ce chapitre.

National Instruments supporte les anciennes fonctions utilisant des fonctions de compatibilité, mais toutes les nouvelles applications doivent être élaborées avec les nouvelles fonctions. Le tableau suivant présente la correspondance entre les anciennes et les nouvelles fonctions.

Tableau 51-1. Anciennes et nouvelles fonctions d'Automation ActiveX

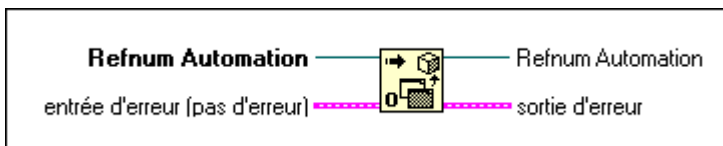
Nouvelles fonctions ActiveX	Anciennes fonctions ActiveX
Automation Open	Create Refnum
Automation Close	Release Refnum
Nœud de méthode	Execute Method
Nœud de propriété	Get property Set property

Descriptions des fonctions d'Automation ActiveX

Les fonctions suivantes sont disponibles.

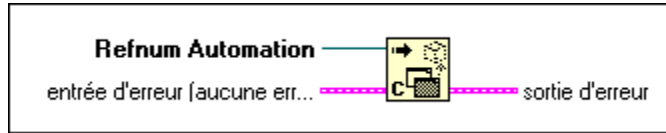
Ouvrir refnum Automation

Ouvre un refnum d’automation qui se rapporte à un objet Automation ActiveX spécifique. Vous pouvez sélectionner la classe de l’objet en ouvrant le menu local de la fonction et en sélectionnant **Sélectionner une classe ActiveX**. Une fois que vous ouvrez un refnum, vous pouvez le transmettre à d’autres fonctions ActiveX. Ne sélectionnez que des classes valides comme entrées de cette fonction.



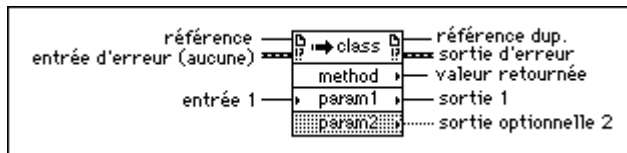
Fermer refnum Automation

Ferme un refnum d'automation. Assurez-vous d'avoir fermé tous les refnums d'automation ouverts lorsque vous n'en avez plus besoin.



Nœud de méthode

Invoke une méthode ou action sur un objet ActiveX. Pour sélectionner une classe d'objet ActiveX, ouvrez le menu local et sélectionnez **Sélectionner une classe ActiveX** ou câblez un refnum d'automation à l'entrée. Pour sélectionner une méthode associée à cet objet, ouvrez le menu local de la seconde section du nœud ("méthode" dans le diagramme) et sélectionnez **méthodes**. Une fois que vous avez sélectionné la méthode, les paramètres associés apparaissent en dessous. Vous pouvez lire ou écrire les valeurs des paramètres. Les paramètres sur fond blanc sont des entrées requises et les paramètres sur fond gris sont des entrées optionnelles.

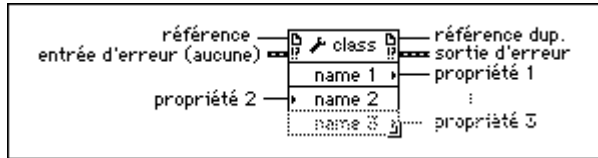


Si les paramètres d'entrées sont de type variant, vous pouvez leur câbler des types de données G qui seront automatiquement convertis en types de données variant et repérés par un point de coercion. Si une sortie est du type variant, utilisez la fonction "En données G" pour la convertir en type de données G, si nécessaire.

Nœud de propriété

Définit (écrit) ou obtient (lit) les informations relatives aux propriétés d'un objet ActiveX. Pour sélectionner une classe d'objet ActiveX, ouvrez le menu local et sélectionnez **Sélectionner une classe ActiveX** ou câblez un refnum d'automation en entrée. Pour sélectionner une propriété associée à cet objet, ouvrez le menu local de la seconde ligne du nœud et sélectionnez **propriétés**. Pour définir les informations relatives aux propriétés, ouvrez le menu local et sélectionnez **Changer en écriture**, et pour obtenir des informations relatives à la propriété, ouvrez le menu local et sélectionnez **Changer en lecture**. Certaines propriétés sont à écriture ou lecture uniquement et l'affichage de **Changer en écriture** ou **Changer en lecture** est grisé dans le menu local.

Le fonctionnement du Nœud de propriété est comparable à celui de l'Attribute Node. Si vous souhaitez ajouter des éléments au nœud, ouvrez le menu local et sélectionnez **Ajouter un élément** ou cliquez et déplacez le nœud pour étendre son nombre d'éléments. Les propriétés sont modifiées de haut en bas. Souvenez-vous que si la petite flèche de direction d'une propriété se trouve à gauche, vous définissez la valeur de la propriété. Si elle se trouve à droite, vous obtenez la valeur de la propriété.



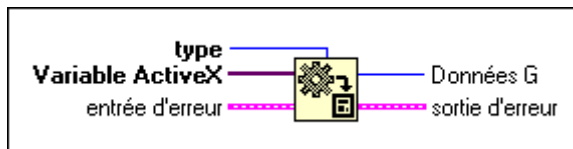
Si la propriété à écrire est du type Variable ActiveX, vous pouvez la câbler aux types de données G qui seront automatiquement convertis en types de données variant et repéré par un point de coercion. Si la propriété est du type Variable ActiveX, utilisez la fonction "En données G" pour la convertir en type G, si nécessaire.

Fonctions de conversion de données

Certaines applications fournissent des données ActiveX sous forme d'un type de données autodéscriptif nommé *ActiveX* ou *Variable OLE*. Pour revoir les données ou les traiter en G, vous devez les convertir au type de données V correspondant. Pour convertir des données du type Variable ActiveX en données G, utilisez la fonction "En données G" décrite ci-dessous.

En données G

Convertit les données du type Variable ActiveX en données pouvant s'afficher sous LabVIEW.



Vis AppleEvent



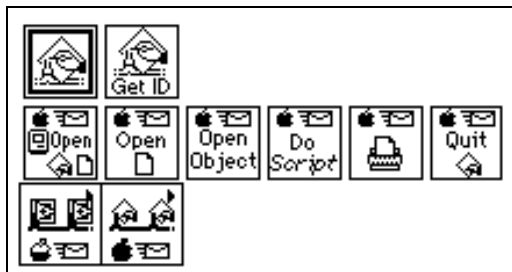
Remarque *Ce chapitre ne s'applique qu'aux utilisateurs de LabVIEW sur la plate-forme Macintosh System 7.*

Ce chapitre décrit les Vis LabVIEW d'AppleEvents, une forme de communication interapplication (IAC), grâce à laquelle les applications Macintosh communiquent entre elles. Vous pouvez également utiliser LabVIEW avec une forme d'IAC de bas niveau dite communication entre programmes (PPC).

Les AppleEvents représentent une méthode de communications de niveau supérieur où des applications utilisent des messages pour demander à d'autres applications de réaliser certaines actions ou de retourner des informations. Une application peut envoyer ces messages à elle-même, à d'autres applications sur une même machine ou à d'autres applications n'importe où sur le réseau. Apple a défini un *vocabulaire* de messages standardisant cette forme de communications interapplication. Vous pouvez combiner des *mots* dans ce vocabulaire pour former des messages complexes. Ce vocabulaire est décrit en détail dans *AppleEvent Registry*, un document disponible auprès d'Apple Computer, Inc. La plupart des applications écrites pour le System 7, notamment LabVIEW, réagissent au même sous-ensemble d'AppleEvents.

PPC est une forme d'IAC de bas niveau par lequel les applications envoient et reçoivent des blocs de données. PPC offre une meilleure performance que celle d'AppleEvents, car les délais requis par la transmission sont moindres. Cependant, vu que le PPC ne définit pas les types d'informations transmises, beaucoup d'applications ne le supportent pas. PPC est la meilleure façon d'envoyer de gros volumes d'informations entre applications qui supportent PPC. Reportez-vous au chapitre 53, *Vis de communication entre programmes*, pour plus d'informations sur PPC.

L'illustration suivante présente la palette **VI AppleEvent**, à laquelle vous pouvez accéder en sélectionnant **Fonctions»Communication»AppleEvent**



Remarque *Pour que les applications puissent communiquer avec l'IAC, l'ordinateur doit utiliser System 7.0, ou une version ultérieure, avec Program Linking (liaison de programmes) activé.*

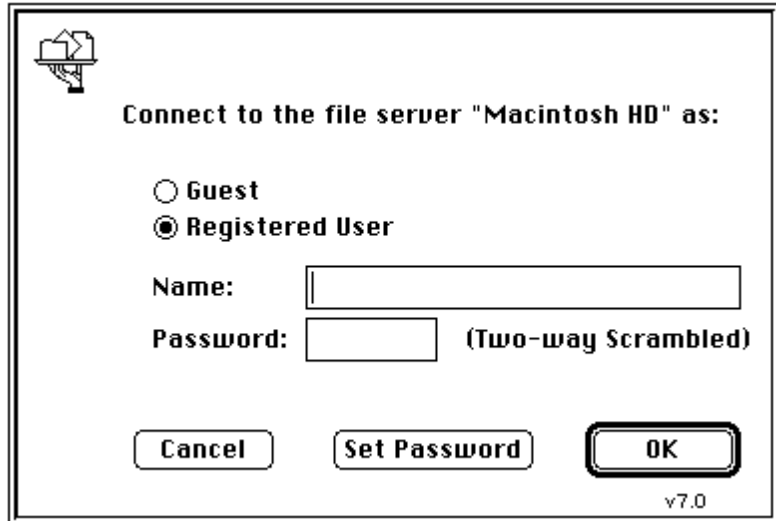
Pour des exemples sur l'utilisation des VIs AppleEvent, consultez la bibliothèque `examples\comm\AE Examples.11b`.

Comportement général des VIs AppleEvent

Lorsque vous envoyez un AppleEvent, vous devez spécifier l'application *destination* pour l'événement. Pour recevoir un AppleEvent, l'application destination doit être lancée. Vous pouvez utiliser le VI "AESend Finder Open" pour ouvrir une application.

Boîte de dialogue d'identification de l'utilisateur

Avant d'envoyer un AppleEvent à un autre ordinateur, vous devez ouvrir l'utilitaire du panneau de commandes Utilisateurs et Groupes (User & Group Control Panel) sur l'ordinateur de destination et définir votre nom d'utilisateur et mot de passe. La première fois que vous envoyez un AppleEvent à une application ou Finder de l'ordinateur de destination, une boîte de dialogue vous invite à entrer vos nom et mot de passe. Le système compare ces informations à la configuration de l'utilitaire du panneau de commandes Utilisateurs et Groupes sur l'ordinateur de destination.



La conception actuelle du Gestionnaire AppleEvent ne comporte pas de méthode programmatique pour contourner cette boîte de dialogue. Pensez-y donc lorsque vous élaborez des VIs qui se servent de l'IAC. Par exemple, vous ne pouvez pas demander qu'un ordinateur à distance sans surveillance envoie un AppleEvent à un troisième ordinateur ; quelqu'un doit entrer les informations dans la boîte de dialogue d'Identification de l'utilisateur qui apparaît sur l'écran de l'ordinateur à distance. Les VIs PPC permettent des sessions *non établies* si un accès hôte est activé sur l'ordinateur avec lequel vous souhaitez communiquer. Les VIs PPC sont ainsi plus utiles pour certains types de communications de LabVIEW-à-LabVIEW.

Identification de la destination

La plupart des VIs qui envoient des AppleEvents exigent une description de l'application destination qui reçoit l'AppleEvent. Le paramètre **ID de la destination** est un cluster complexe d'informations, défini par Apple Computer Inc., décrivant l'application destination et son emplacement. Les VIs suivants génèrent la valeur **ID de la destination** et vous n'avez donc pas besoin de créer ce cluster sur le diagramme.

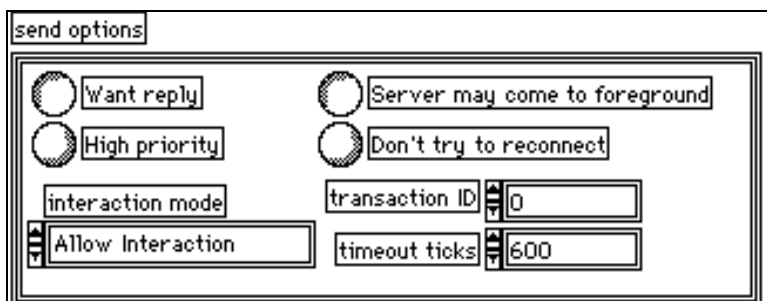
- “PPC Browser” crée l'**ID de la destination** en affichant une boîte de dialogue où vous pouvez interactivement sélectionner des applications sensibles à AppleEvent sur le réseau.
- “Get Target ID” crée l'**ID de la destination** programmatiquement à partir du nom de l'application et de son emplacement sur le réseau.

Ces VIs sont traités plus en détails dans la section *Descriptions des VIs de ciblage*, ci-dessous.

Ne considérez le cluster **ID de la destination** que si vous souhaitez transmettre des informations de destination d'un VI à un autre. Pour créer un cluster **ID de la destination** pour la face-avant d'un VI qui transmette des informations à un autre VI ou à un AppleEvent, vous pouvez copier le cluster **ID de la destination** de la face-avant d'un des VIs AppleEvent.

Options d'envoi

La plupart des VIs qui envoient un AppleEvent comprennent une entrée **options d'envoi**, qui spécifie si l'application destination peut interagir avec l'utilisateur ainsi que la longueur du timeout AppleEvent.

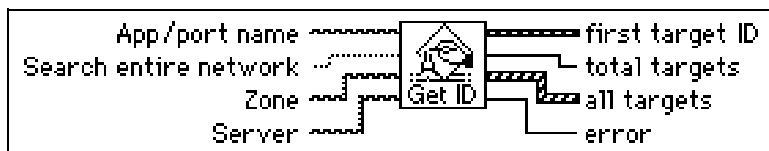


Descriptions des VIs de ciblage

Les VIs de ciblage suivants sont disponibles.

Get Target ID

Retourne l'identification d'une destination pour une application spécifiée selon son nom et emplacement. Vous pouvez spécifier le nom ou l'emplacement de l'application, sinon le VI cherche l'application sur tout le réseau.

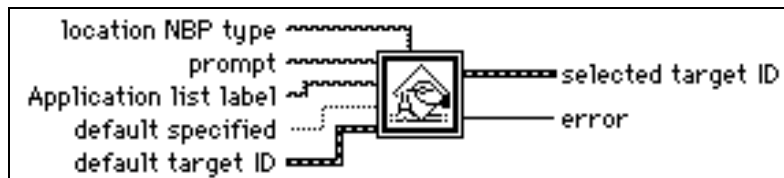


Le tableau suivant présente un résumé des opérations **Chercher sur tout le réseau**, **Zone** et **Serveur** :

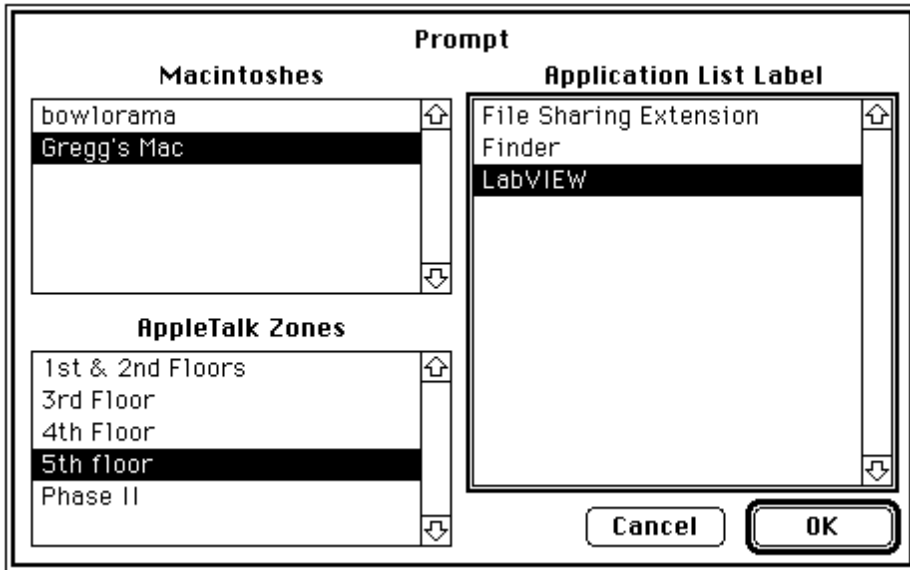
Pour chercher les emplacements suivants :	Utilisez les paramètres suivants
L'ordinateur courant	Zone et Serveur ne doivent pas être câblés. Chercher sur tout le réseau doit être FALSE.
Un ordinateur spécifié sur le réseau	Zone et Serveur doivent spécifier la zone et le serveur de l'ordinateur de destination. (Si vous ne câblez pas Zone , le VI cherche dans la zone courante.) Chercher sur tout le réseau doit être FALSE.
Une zone spécifiée	Zone doit spécifier la zone à chercher. Serveur ne doit pas être câblé. Chercher sur tout le réseau doit être FALSE.
Tout le réseau	Chercher sur tout le réseau doit être TRUE. Le VI ignore Zone et Serveur .

PPC Browser

Invoque la boîte de dialogue PPC Browser pour sélectionner l'application sur un réseau ou sur un même ordinateur.



Vous pouvez utiliser cette boîte de dialogue standard de Macintosh pour sélectionner la zone d'un réseau, un objet dans cette zone (sous System 7, c'est en général le nom de l'ordinateur de l'utilisateur) et une application. Ce VI retourne ensuite le cluster **ID de la destination**.

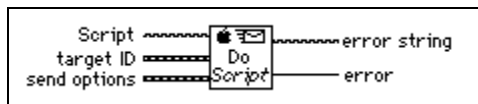


Descriptions des VIs AppleEvent

Les VIs AppleEvent suivants sont disponibles.

AESend Do Script


Envoie l'AppleEvent Do Script à l'application de destination spécifiée.



AESend Finder Open

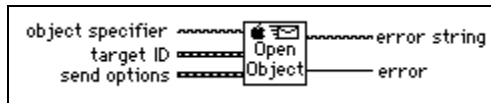
Envoie l'AppleEvent pour ouvrir des applications ou des documents spécifiés au Finder du System 7 de la machine spécifiée.



 **Remarque** *Apple peut changer l'ensemble d'AppleEvents auquel le Finder réagit pour qu'il se rapproche le plus possible de l'ensemble standard d'AppleEvents. Il est donc possible que l'AppleEvent envoyé par AESend Finder Open au Finder ne soit pas supporté par de futures versions du logiciel système.*

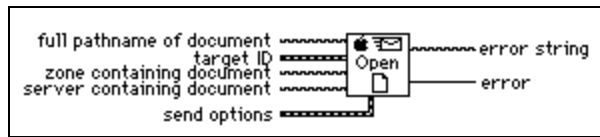
AESend Open

Envoie l'AppleEvent Open à l'application de destination spécifiée.



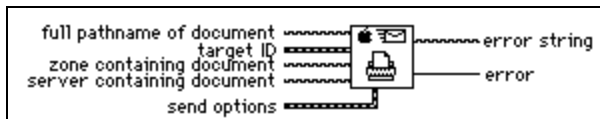
AESend Open Document

Envoie l'AppleEvent Open Document à l'application de destination spécifiée, lui indiquant d'ouvrir le document spécifié.



AESend Print Document

Envoie l'AppleEvent Print Document à l'application de destination spécifiée, lui indiquant d'imprimer le document spécifié.



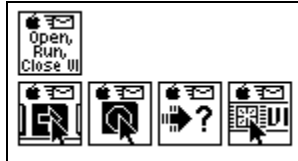
AESend Quit Application

Envoie l'AppleEvent Quit Application à l'application de destination spécifiée.



VIs AppleEvent spécifiques à LabVIEW

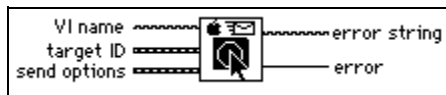
Les VIs AppleEvent spécifiques à LabVIEW envoient des messages que seules les applications LabVIEW peuvent reconnaître (systèmes standard et d'exécution). Pour accéder aux VIs AppleEvent spécifiques à LabVIEW, sélectionnez **Fonctions»Communication»AppleEvent spécifique à LabVIEW**.



N'utilisez ces VIs que lorsque vous communiquez avec des applications LabVIEW. Vous pouvez envoyer ces messages à l'application LabVIEW courante ou à une application LabVIEW sur un réseau. Reportez-vous au tableau A-5, *Codes d'erreur AppleEvents*, de l'Annexe A, *Codes d'erreur*, pour des informations sur les erreurs.

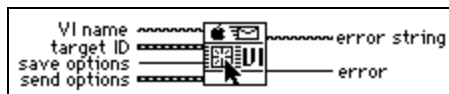
AESend Abort VI

Envoie l'AppleEvent Abort VI à l'application LabVIEW de destination spécifiée.



AESend Close VI

Envoie l'AppleEvent Close VI à l'application LabVIEW de destination spécifiée.



AESend Open, Run, Close VI

Utilise les VIs AppleEvent Open Document, Run VI, VI Active? et Close VI pour faire en sorte qu'une application LabVIEW spécifiée ouvre, exécute et ferme un VI.



Pour utiliser ce VI, vous devez spécifier le nom de chemin complet du VI que vous souhaitez exécuter. Reportez-vous au chapitre 12, *Indicateurs et commandes du chemin et du refnum*, de votre *Manuel de référence de programmation en G* pour une description des commandes et indicateurs de chemin disponibles dans la palette Commandes.

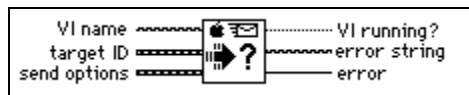
AESend Run VI

Envoie l'AppleEvent Run VI à l'application LabVIEW de destination spécifiée.



AESend VI Active?

Envoie l'AppleEvent VI Active? à l'application LabVIEW de destination spécifiée. Le paramètre **VI en cours d'exécution?** est un booléen indiquant si le VI est en cours d'exécution.



Sujets avancés

Cette section décrit quelques techniques de programmation avancées que vous pouvez mettre en oeuvre avec les VIs AppleEvent.

Construire et envoyer d'autres AppleEvents

En plus des VIs qui envoient des AppleEvents communs, vous pouvez utiliser des VIs de niveau inférieur pour envoyer n'importe quel AppleEvent. Ces VIs exigent des connaissances plus approfondies que les VIs décrits plus haut dans ce chapitre. Si vous souhaitez utiliser ces VIs, familiarisez-vous avec les informations concernant AppleEvents figurant dans *Inside Macintosh, Volume VI* et dans le document *AppleEvent Registry*.

Lorsque vous envoyez un AppleEvent, vous devez inclure plusieurs renseignements. La classe d'événement et l'ID de l'événement identifient l'AppleEvent que vous envoyez. La classe d'événement est un code à quatre lettres qui identifie le groupe AppleEvent. Par exemple, une classe d'événement `core` identifie un AppleEvent comme appartenant à l'ensemble d'AppleEvents central. L'événement ID est un autre code à quatre lettres qui identifie l'AppleEvent précis que vous souhaitez envoyer.

Par exemple, `odoc` est le code à quatre lettres pour l'AppleEvent "Open documents", un des AppleEvents centraux. Pour envoyer un AppleEvent par le biais du VI `AESend`, concaténez la classe d'événement et l'ID d'événement en une chaîne de huit caractères. Par exemple, pour envoyer l'AppleEvent "Open documents", transmettez le code à huit caractères `coreodoc` au VI `AESend`.

Si vous envoyez l'AppleEvent à une autre application, vous devez spécifier l'**ID de la destination** et les **options d'envoi**, comme le décrit le chapitre précédent.

Vous pouvez également spécifier un tableau de paramètres si l'application de destination nécessite des informations supplémentaires pour exécuter l'AppleEvent spécifié. Etant donné que la structure de données pour les paramètres AppleEvent ne convient pas aux diagrammes LabVIEW, le VI `AESend` accepte ces paramètres comme chaînes ASCII. Ces chaînes doivent suivre la grammaire décrite dans la section ci-après. Vous pouvez utiliser cette grammaire pour décrire n'importe quel paramètre AppleEvent. Le VI `AESend` interprète cette chaîne pour créer la structure de données appropriée d'un AppleEvent et envoie ensuite l'événement à la destination spécifiée.

Créer des paramètres AppleEvent

Souvent, un paramètre AppleEvent représente une valeur unique ; cependant, il peut être assez complexe, pourvu d'une structure hiérarchique composée d'éléments qui à leur tour comprennent d'autres éléments. Dans LabVIEW, un paramètre est élaboré comme chaîne de caractères, dont la grammaire simple vous permet de décrire tous les types de données qu'un paramètre AppleEvent peut représenter, notamment des structures complexes.

Une chaîne de caractères d'un paramètre AppleEvent commence par un mot clé, un code à quatre lettres décrivant la signification du paramètre. Par exemple, si le paramètre est un paramètre direct (un des types de paramètres les plus communs), vous devez spécifier que le mot clé est un `keyDirectObject` (objet direct clé) en utilisant le code à quatre lettres `----` (quatre tirets). Un autre exemple de mot clé est `savo`, abréviation de "sauvegarder des options", qui s'utilise lorsque l'AppleEvent "Close VI" est envoyé à LabVIEW. La documentation

traitant des AppleEvents supportés par l'application doit contenir les mots clés de chaque paramètre. Reportez-vous à la section *Envoyer des AppleEvents à LabVIEW depuis d'autres applications* dans ce chapitre pour obtenir une liste des AppleEvents que vous pouvez utiliser avec LabVIEW.

Après le mot clé, vous devez spécifier les données du paramètre sous forme de chaîne de caractères. Vous pouvez utiliser les AppleEvents avec différents types de données, y compris des chaînes de caractères et des nombres. Lorsque vous spécifiez une chaîne de caractères des données, l'AppleEvent AESend VI la convertit en un type de données souhaité selon le format des données et les directives optionnelles intégrées dans la chaîne de caractères. Chaque élément des données comporte un code à quatre mots associé, qui indique son type de données. L'application de destination utilise ce code pour interpréter les données. Par exemple, si des éléments séparés par des virgules se trouvent entre crochets, une liste de *Descripteurs d'AE* est créée comprenant le type de données de `list` (liste) ; chacun des éléments séparés par des virgules peut à son tour être un autre élément, y compris des listes.

Vous pouvez utiliser plusieurs VIs dans la palette **VI AppleEvents** pour créer quelques-unes des chaînes de caractères de paramètres les plus communes, notamment les pseudonymes, utiles lorsque des fichiers sont référencés en paramètres et les listes de descripteurs, qui permettent de lister des éléments en tant que paramètre. Vous pouvez concaténer ou cascader ces chaînes de caractères pour créer un paramètre plus complexe.

Le tableau 52-1 décrit le format des chaînes de caractères descriptives d'AppleEvent et indique quels sont les VIs qui peuvent créer le descripteur, le cas échéant.

Tableau 52-1. Formats des chaînes de caractères descriptives d'AppleEvent

Pour envoyer des données sous la forme	Formate la chaîne de caractères sous la forme	Type de code du paramètre	Exemples	VI pouvant construire la chaîne de caractères
d'entier	d'une série de chiffres décimaux, précédée d'un signe de soustraction (en option).	long ou court	1234 -5678	sans objet
de données énumérées	d'un code à quatre lettres. S'il est trop long, il est tronqué ; s'il est trop court, il est comblé par des espaces. Si vous l'entourez d'une apostrophe ('), il peut contenir n'importe quel caractère ; sinon, il ne peut pas contenir : @ ' : - , ([{ }]) et ne peut pas commencer par un chiffre.	enum	whos '@all' long >= '86it'	sans objet
de chaîne de caractères	de la séquence de caractères souhaitée entre guillemets ("entré avec <option-[> et " entré avec <option-Maj-[>). Notez que la chaîne de caractères n'est pas à terminaison nulle.	TEXT	"put x into card field 5" "Hi There"	sans objet
d'une entrée AE	d'une liste d'éléments séparés par des virgules entre accolades où chaque élément consiste en un mot clé (un code type) suivi par deux-points, suivis par une valeur, qui peut être de n'importe quel type listé dans cette table.	reco	{x:100, y:-100} {'origin': {x:100, y:- 100}, extent: {x:500, y:500}, cont:[1,5,25]}	AECreat Record

Tableau 52-1. Formats des chaînes de caractères descriptives d'AppleEvent (Suite)

Pour envoyer des données sous la forme	Formate la chaîne de caractères sous la forme	Type de code du paramètre	Exemples	VI pouvant construire la chaîne de caractères
d'une liste de descripteurs AE	d'une liste de descripteurs séparés par des virgules entre crochets.	list	[123, -58, "test"]	AECreate Descriptor List
de données hexa	d'un nombre pair de chiffres hexa entre doubles chevrons («entré avec <option-\> et » entré avec <option-Maj-\>).	?? (doit être contraint – reportez-vous à l'élément suivant)	«01 57 64 fe AB C1»	(Données hexa sont une composante de la chaîne de caractères produite par le VI Make Alias)
d'un autre type de données	de données intégrées créées dans l'un des types listés dans ce tableau entre parenthèses avec le code de type souhaité mis en face. Si les données sont numériques, LabVIEW les contraint dans le type spécifié si possible et retourne le code d'erreur <code>errAECOercionFail</code> s'il ne le peut pas. Si les données sont d'un type différent, LabVIEW remplace l'ancien code de type avec le code de type spécifié.	le code de type spécifié	<pre> sing(1234) alis («hex dump de un Pseudonyme») type(line) rang{star: 5, stop: 6} </pre>	<p>sans objet VI Make Alias crée un hexdump d'une description du fichier.</p> <p>sans objet sans objet</p>
de données nulles	de chaîne de caractères vide contrainte en type nul.	nul	()	sans objet

VIs AppleEvent de niveau inférieur

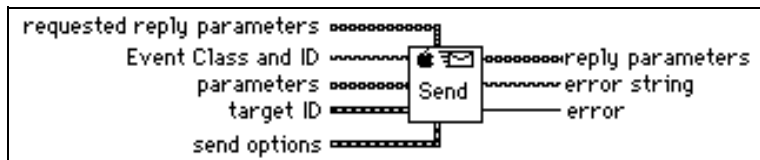
Vous pouvez utiliser les VIs AppleEvent de niveau inférieur pour construire des paramètres AppleEvent et envoyer l'AppleEvent. Les VIs de niveau supérieur utilisés pour envoyer des AppleEvents et décrits plus haut dans ce chapitre reposent sur le VI AESend et illustrent la création d'AppleEvents et de leurs paramètres.

Pour accéder à la palette **AppleEvent de niveau inférieur**, cliquez sur l'icône **AppleEvent de niveau inférieur**.



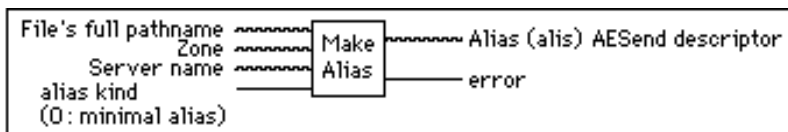
AESend

Envoie un AppleEvent spécifié dans les paramètres à l'application de destination spécifiée.



Make Alias

Crée une description unique d'un fichier en se basant sur son nom de chemin et son emplacement sur le réseau. Vous pouvez utiliser cette description avec le VI AESend lorsque vous envoyez un AppleEvent qui se rapporte à un fichier.



Un "Alias" est une structure de données utilisée par la boîte à outils de Macintosh pour décrire des objets de système de fichiers (fichiers, répertoires et volumes). Ne le confondez pas avec un fichier alias Finder. Un alias minimal comporte un nom de chemin vers le fichier et éventuellement la zone et le serveur où le fichier réside. Un alias complet comprend plus d'informations, telles que la date de création, le type de fichier et le créateur. (La description complète de la structure d'un alias est confidentielle et propriété d'Apple Computer). Les

alias sont la façon la plus commune de spécifier un objet de système de fichier comme paramètre à un AppleEvent.

Créer des paramètres AppleEvent avec des spécificateurs d'objets

Apple a élaboré une interface de niveau supérieur pour créer des AppleEvents, nommée Bibliothèque de support d'objets (Object Support Library). Cette interface est une couche qui repose sur les structures de données des paramètres AppleEvent décrites antérieurement dans ce chapitre. Cette interface aide à créer des types communs de paramètres, y compris les spécifications de gamme. Les VIs de support d'objets LabVIEW se trouvent sur la palette **AppleEvent de niveau inférieur**.

AECreat Comp Descriptor

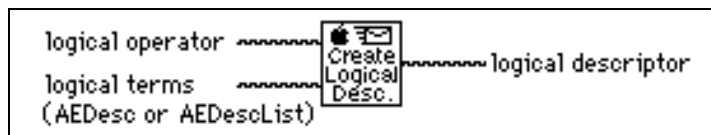
Crée une chaîne de caractères décrivant un enregistrement de comparaison AppleEvent, qui spécifie comment comparer des objets AppleEvent avec un autre objet AppleEvent ou un enregistrement de descripteur.



Vous pouvez utiliser par exemple la chaîne de caractères de sortie du descripteur de comparaison comme argument du VI AESend, ou comme argument du VI AECreat Object Specifier pour construire une chaîne de caractères descriptive plus complexe. Reportez-vous à la section *Exemple de VI de support d'objets* dans ce chapitre pour un exemple sur son utilisation.

AECreat Logical Descriptor

Crée une chaîne de caractères décrivant un descripteur logique AppleEvent, que vous pouvez utiliser avec le VI AESend.



Les enregistrements logiques AppleEvent décrivent des expressions logiques ou booléennes de plusieurs termes, tels que le AND de deux enregistrements de comparaison AppleEvent. Par exemple, vous pouvez utiliser la chaîne de caractères de sortie du descripteur logique du VI AESend, ou comme argument du VI AECreat Object Specifier pour construire une chaîne de caractères descriptive plus complexe. Reportez-vous à la section *Exemple de VI de support d'objets* dans ce chapitre pour un exemple sur son utilisation.

AECreat Object Specifier

Crée une chaîne de caractères décrivant un objet, que vous pouvez utiliser avec le VI AESend.



Un spécificateur d'objet est un enregistrement AppleEvent de type `obj` qui décrit un objet précis. Il est composé de quatre éléments : la classe de l'objet, l'objet conteneur, un code indiquant la forme de la description et la description de l'objet.

AECreat Range Descriptor

Crée une chaîne de caractères décrivant un enregistrement descripteur de gamme AppleEvent, que vous pouvez utiliser avec le VI AESend.



Les enregistrements de descripteurs de gamme sont utilisés dans des spécificateurs d'objets dont la forme clé est `formRange` (`rang`). Ils décrivent une gamme d'objets avec deux spécificateurs d'objets : le début et la fin de la gamme.

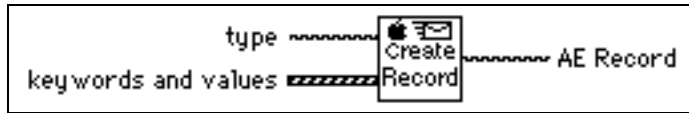
AECreat Descriptor List

Crée une chaîne de caractères décrivant une liste de descripteurs AppleEvent, que vous pouvez utiliser ensuite avec le VI AESend. En général, vous devez utiliser les listes de descripteurs lorsque vous créez des opérandes pour un descripteur logique.



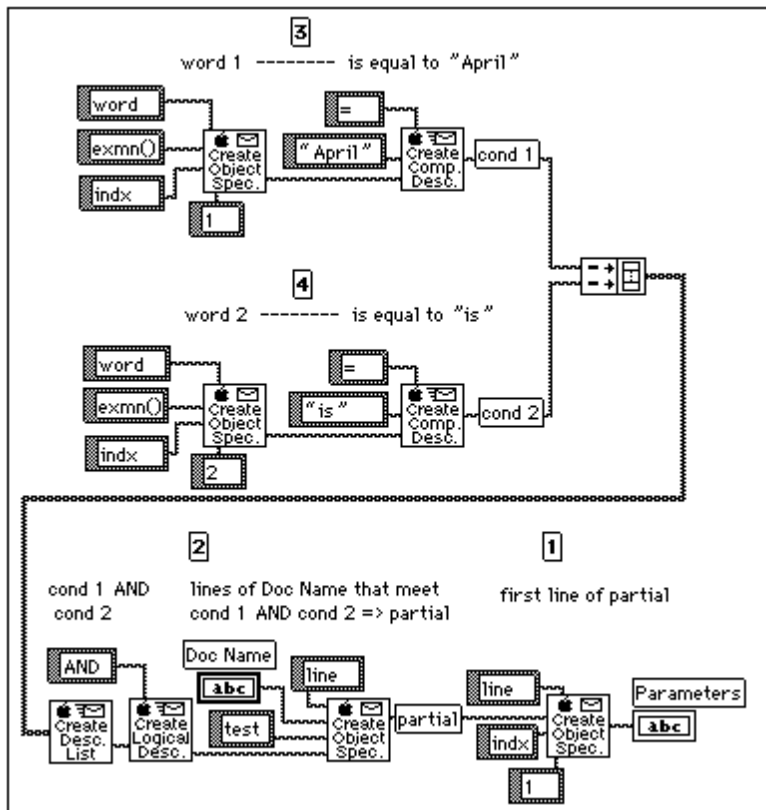
AECreat Record

Crée une chaîne de caractères décrivant un enregistrement descripteur AppleEvent, que vous pouvez ensuite utiliser avec le VI AESend. Vous pouvez utiliser un descripteur d'enregistrement pour grouper des descripteurs de divers types. Chaque descripteur comporte son propre mot clé, ou nom et valeur.



Exemple de VI de support d'objets

L'exemple suivant crée un paramètre AppleEvent par le biais des VIs de support d'objet. Cet exemple crée un paramètre AppleEvent à envoyer à un système de traitement de texte, lui dictant de retourner la première ligne d'un document spécifié dont le premier mot est **april** et le second est **is**.



La chaîne de caractères suivante créée par le diagramme précédent est assez complexe ; des tabs sont ajoutés pour faciliter la lecture de la chaîne de caractères. Pour plus d'informations concernant la Bibliothèque de support d'objets, consultez le document *AppleEvent Registry*.

```
obj
  want: type('line'),
  from: obj {
    want: type('line'),
    from: Doc Name,
    form: test,
    seld: logi {
      term:[
        cmpd{
          relo:=,
          obj1:"April",
          obj2:obj {
            want: type('word'),
            from: exmn( ),
            form: indx,
            seld: 1
          }
        },
        cmpd{
          relo:=,
          obj1:"is",
          obj2:obj {
            want: type('word'),
            from: exmn( ),
            form: indx,
            seld: 2
          }
        }
      ],
      logc: AND
    }
  },
  form: indx,
  seld: 1
```

Envoyer des AppleEvents à LabVIEW depuis d'autres applications

LabVIEW réagit aux AppleEvents requis, dont Apple a prévu le support par toutes les applications de System 7 et aux AppleEvents spécifiques de LabVIEW, conçus spécifiquement pour LabVIEW. Ces deux catégories sont décrites dans les sections qui suivent.

AppleEvents requis

LabVIEW réagit aux AppleEvents requis, qui sont “Ouvrir une application” (Open Application), “Ouvrir des documents” (Open Documents), “Imprimer des documents” (Print Documents) et “Quitter une application” (Quit Application). Ces événements sont décrits dans *Inside Macintosh, Volume VI*.

AppleEvents spécifiques à LabVIEW

LabVIEW réagit également aux AppleEvents spécifiques à LabVIEW Open Document, Run VI, VI Active? et Close VI. Vous pouvez, avec ces événements et l'AppleEvent Open Application, utiliser d'autres applications pour demander à LabVIEW, de façon programmatique, d'ouvrir un VI, de l'exécuter et de le fermer quand il s'achève. De solides connaissances sur les AppleEvents, comme décrit dans *Inside Macintosh, Volume VI*, et sur les registres d'AppleEvent sont essentielles si vous souhaitez envoyer ces AppleEvents à LabVIEW depuis d'autres applications. Vous pouvez envoyer ces événements entre plusieurs applications LabVIEW en utilisant les VIs Utilitaires décrits dans la section *Envoyer des AppleEvents* au chapitre 24, *AppleEvents*, du *Manuel de l'utilisateur LabVIEW*.

Les AppleEvents spécifiques à LabVIEW sont décrits dans des sections ultérieures, avec un format comparable à celui des registres d'AppleEvents.

Réponses aux AppleEvents

Si LabVIEW ne peut pas exécuter un AppleEvent, la réponse contient un code d'erreur. Si ce n'est pas une erreur standard d'AppleEvent, la réponse contient également une chaîne de caractères décrivant l'erreur. L'Annexe A, *Codes d'erreur*, présente un résumé des erreurs spécifiques à LabVIEW pouvant être retournées en réponse à un AppleEvent.

Événement : Run VI

Description

Indique à LabVIEW d'exécuter le ou les VI(s) spécifié(s). Avant d'exécuter cet événement, l'application LabVIEW doit être lancée et le VI doit être ouvert (vous pouvez ouvrir le VI en utilisant l'AppleEvent Open Documents).

Classe d'événement

LBVW (Les événements personnalisés utilisent le type de créateur d'application pour la classe d'événement.)

ID de l'événement

GoVI ----

Paramètres de l'événement

Description	Mot clé	Type par défaut
VI ou Liste de VIs	keyDirectObject (----)	typechar (car.) (requis) ou liste de typechar (liste)

Paramètres de réponse

Description	Mot clé	Type par défaut
aucune		

Erreurs possibles

Erreur	Valeur	Description
kLVE_InvalidState	1000	Le VI est dans un état qui ne permet pas son exécution.
kLVE_FPNotOpen	1001	La face-avant du VI n'est pas ouverte.
kLVE_CtrlErr	1002	Certaines commandes de la face-avant du VI sont dans un état d'erreur.
kLVE_VIBad	1003	Le VI est brisé.
kLVE_NotInMem	1004	Le VI ne réside pas en mémoire.

Événement : Abort VI

Description

Indique à LabVIEW d'annuler l'exécution des VIs spécifiés. Avant d'exécuter cet événement, l'application LabVIEW doit être lancée et les VIs doivent être ouverts (vous pouvez ouvrir les VIs en utilisant l'AppleEvent "Ouvrir des documents" (Open Documents)). Ce message ne peut être envoyé qu'à des VIs exécutés depuis le niveau supérieur (les sous-VIs ne sont annulés que si le VI appelant l'est aussi).

Classe d'événement

LBVW (Les événements personnalisés utilisent le type de créateur d'application pour la classe d'événement.)

ID de l'événement

RsVI

Paramètres de l'événement

Description	Mot clé	Type par défaut
VI ou Liste de VIs	keyDirectObject (----)	typechar (car.) (requis) ou liste de typechar (liste)

Paramètres de réponse

Description	Mot clé Requis ?	Type par défaut
aucune		

Erreurs possibles

Erreur	Valeur	Description
kLVE_InvalidState	1000	Le VI est dans un état qui ne permet pas son exécution.
kLVE_FPNotOpen	1001	La face-avant du VI n'est pas ouverte.
kLVE_NotInMem	1004	Le VI ne réside pas en mémoire.

Événement : VI Active?

Description

Demande si un VI spécifié est en cours d'exécution. Avant d'exécuter cet événement, l'application LabVIEW doit être lancée et le VI doit être ouvert (vous pouvez ouvrir le VI en utilisant l'AppleEvent "Ouvrir des documents" (Open Documents)). La réponse indique si le VI est en cours d'exécution.

Classe d'événement

LBVW (Les événements personnalisés utilisent le type de créateur d'application pour la classe d'événement.)

ID de l'événement

VIAC

Paramètres de l'événement

Description	Mot clé	Type par défaut
Nom du VI (requis)	keyDirectObject (----)	typechar (car.)

Paramètres de réponse

Description	Mot clé	Type par défaut
Actif? (requis)	keyDirectObject (----)	typeBoolean (bool)

Erreurs possibles

Erreur	Valeur	Description
kAEvtErrFPNotOpen	1001	La face-avant du VI n'est pas ouverte.
kLVE_NotInMem	1004	Le VI ne réside pas en mémoire.

Événement : Close VI

Description

Indique à LabVIEW de fermer le ou les VI(s) spécifié(s). Avant d'exécuter cet événement, l'application LabVIEW doit être lancée et le VI doit être ouvert (vous pouvez ouvrir le VI en utilisant l'AppleEvent "Ouvrir des documents" (Open Documents)).

Classe d'événement

LBVW (Les événements personnalisés utilisent le type de créateur d'application pour la classe d'événement.)

ID de l'événement

ClVI

Paramètres de l'événement

Description	Mot clé	Type par défaut
VI ou Liste de VIs	keyDirectObject (----)	typechar (car.) (requis) ou liste de typechar (liste)
Options de sauvegarde (non requises)	keyAESaveOptions (savo)	Valeurs possibles Enumtype (enum) : oui et non

Paramètres de réponse

Description	Mot clé	Type par défaut
aucune		

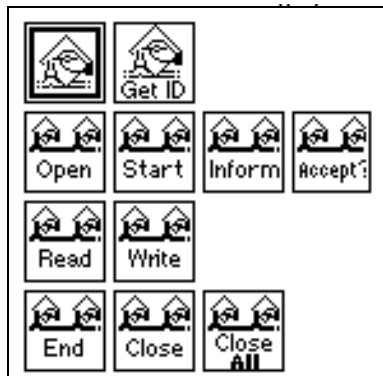
Erreurs possibles

Erreur	Valeur	Description
kAEvtErrrFPNotOpen	1001	La face-avant du VI n'est pas ouverte.
kLVE_NotInMem	1004	Le VI ne réside pas en mémoire.
cancelError	43	L'utilisateur a annulé l'opération Fermer.

VIs de communication entre programmes

Ce chapitre décrit les VIs de LabVIEW pour la communication entre programmes (Program-to-Program Communication [PPC]), une forme de communication interapplication (IAC) de niveau inférieur d'Apple permettant aux applications Macintosh d'envoyer et de recevoir des blocs de données.

L'illustration suivante présente la palette **VI PPC**, que vous pouvez ouvrir en sélectionnant **Fonctions»Communication»PPC**.



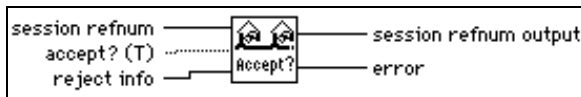
Pour des exemples sur l'utilisation des VIs PPC, consultez la bibliothèque `examples\comm\PPC Examples.llb`.

Descriptions des VIs PPC

Les VIs PPC suivants sont disponibles.

PPC Accept Session

Accepte ou rejette une requête de session PPC selon la valeur booléenne **accepter?**



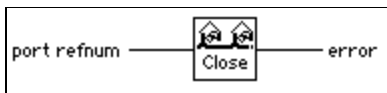
Vous devez accepter ou rejeter la requête en utilisant immédiatement le VI PPC Accept Session, car l'autre ordinateur attend (en état stationnaire) que le VI accepte ou rejette son essai de démarrage de session ou qu'une erreur survienne.

PPC Browser

Pour des informations sur le VI PPC Browser, reportez-vous au chapitre 52, [VIs AppleEvent](#), dans ce manuel.

Close All PPC Ports

Ferme tous les ports PPC ouverts par le VI PPC Open Port.

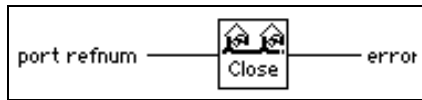


Fermer un port termine tous les appels en cours associés au port avec une erreur "portClosedErr" (erreur -916).

Vous pouvez utiliser le VI Close All PPC Ports pour traiter les conditions anormales qui laissent les ports ouverts. Un exemple de condition anormale peut être généré par un VI annulé avant de pouvoir dûment achever son exécution et fermer le port PPC. Vous pouvez utiliser le VI Close All PPC Ports lors de l'élaboration du VI, quand ce type d'erreurs est le plus commun, ou comme une précaution au début de tout programme qui ouvre des ports.

PPC Close Port

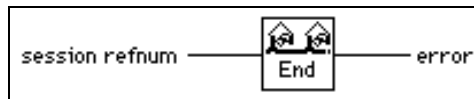
Ferme le port PPC spécifié.



Fermer un port termine tous les appels en cours associés au port par une erreur “portClosedErr” (erreur -916).

PPC End Session

Termine la session PPC spécifiée.



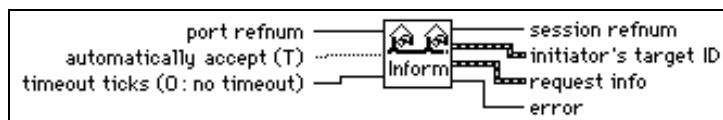
Fermer un port conduit tous les appels en cours associés à la session (appels PPC Read et PPC Write) à se terminer par une erreur “sessClosedErr” (erreur -917).

Get Target ID

Pour des informations sur le VI Get Target ID, reportez-vous au chapitre 52, [Vis AppleEvent](#), dans ce manuel.

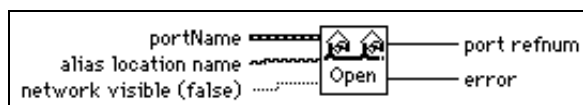
PPC Inform Session

Attend une requête de session PPC.

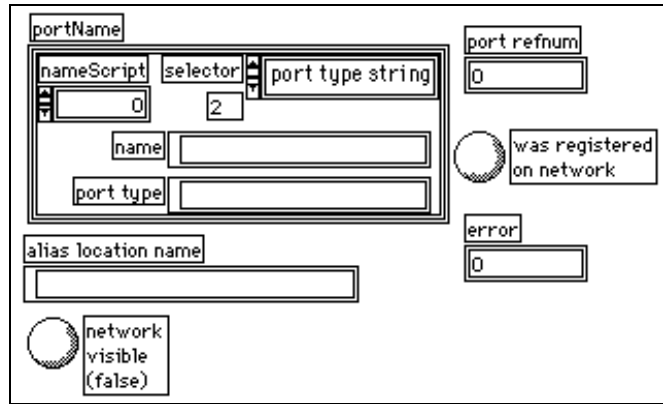


PPC Open Port

Ouvre un port pour des communications PPC et retourne un numéro de référence du port unique dans le paramètre **refnum de port**. Vous pouvez utiliser un port unique pour plusieurs sessions.



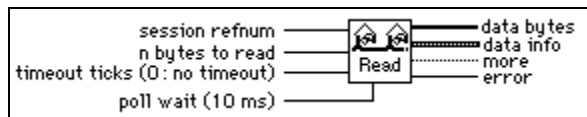
Lorsque vous ouvrez un port avec le VI PPC Open Port, vous devez spécifier un cluster **Nom de port**.



Consultez l'aide en ligne de LabVIEW pour plus d'informations sur ce VI.

PPC Read

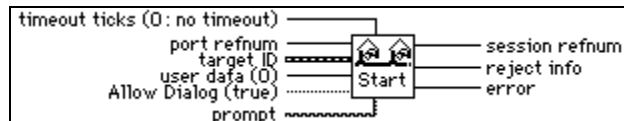
Lit un bloc d'informations d'une session spécifiée. Si un timeout survient ou si l'exécution du VI est annulée avant son achèvement, le port représenté par **refnum de port** est fermé.



Le VI PPC Read s'exécute de manière asynchrone en commençant par lire les données spécifiées et en les interrogeant ensuite jusqu'à la fin de la lecture.

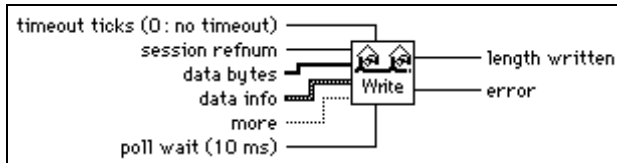
PPC Start Session

Essaie de démarrer une session avec l'application spécifiée par **ID de la destination** à travers le port spécifié. Si un timeout survient ou si l'exécution du VI est annulée avant son achèvement, le port représenté par **refnum de port** est fermé.



PPC Write

Écrit un bloc d'information à la session spécifiée. Si un timeout survient ou si l'exécution du VI est annulée avant son achèvement, le port représenté par **refnum de port** est fermé. Le VI PPC Write s'exécute de manière asynchrone en commençant par écrire les données spécifiées et en les interrogeant ensuite jusqu'à la fin de l'écriture.



Codes d'erreur

Ce document contient des tableaux qui listent tous les codes d'erreur numériques de LabVIEW.

Connectez les VIs Gestionnaires d'erreurs à d'autres VIs pour retourner une description d'une erreur, si elle survient. Les VIs "Gestionnaires d'erreurs" (Error Handler.vi) peuvent également afficher une boîte de dialogue contenant une description de message d'erreur et des boutons pouvant arrêter ou continuer l'exécution. Reportez-vous au sujet *Gestion d'erreurs* dans *L'aide en ligne LabVIEW* pour plus d'informations sur les gestionnaires d'erreurs.



Remarque *Tous les codes d'erreur et descriptions sont également inclus dans les panneaux d'aide de l'utilitaire de configuration sur les plates-formes Windows et Macintosh.*

Codes d'erreur numériques

Les tableaux sont globalement classés dans un ordre croissant des valeurs négatives aux valeurs positives. Les tableaux comportant des valeurs numériques négatives sont classés de la valeur absolue la plus petite à la valeur absolue la plus grande. Remarquez que les codes d'erreur allant de 5000 à 9999 sont destinés à votre utilisation personnelle.

Tableau A-1. Gammes de codes d'erreur numériques

Gamme de codes d'erreur				Tableau
*	-1073807360	à	-1073807231	Codes d'erreur VISA
	-20001	à	-20065	Codes d'erreur des VIs d'analyse
	-10001	à	-10943	Codes d'erreur des VIs d'acquisition de données
	-1700	à	-1719	Codes d'erreur AppleEvent
*	-1200	à	-13xx	Codes d'erreur des drivers d'instruments
	-900	à	-932	Codes d'erreur PPC
*	0	à	85	Codes d'erreur des fonctions de LabVIEW
*	0	à	32	Codes d'erreur GPIB
	1	à	5	Codes d'erreur PPC spécifiques à LabVIEW
*	53	à	66	Codes d'erreur TCP/IP et UDP

Tableau A-1. Gammes de codes d'erreur numériques (Suite)

Gamme de codes d'erreur	Tableau
* 61 à 65	Codes d'erreur port série
1000 à 1004	Codes d'erreur AppleEvent spécifiques à LabVIEW
14001 à 14020	Codes d'erreur DDE

* Ces tableaux contiennent quelques erreurs de valeurs numériques se chevauchant mais ayant des significations différentes, en fonction de la source de l'erreur.

Tableau A-2. Codes d'erreur VISA

Code d'erreur	Nom d'erreur	Description
-1073807360	VI_ERROR_SYSTEM_ERROR	Erreur système inconnue (erreur diverse)
-1073807346	VI_ERROR_INV_OBJECT VI_ERROR_INV_SESSION	La référence de session ou d'objet est invalide
-1073807345	VI_ERROR_RSRC_LOCKED	Un type de verrouillage spécifié ne peut pas être obtenu ou une opération spécifiée ne peut pas être exécutée parce que la ressource est verrouillée
-1073807344	VI_ERROR_INV_EXPR	Expression spécifiée pour la recherche invalide
-1073807343	VI_ERROR_RSRC_NFOUND	Information de position insuffisante ou bien le périphérique ou la ressource ne se trouve pas dans le système
-1073807342	VI_ERROR_INV_RSRC_NAME	Référence de ressource spécifiée invalide. Erreur d'analyse
-1073807341	VI_ERROR_INV_ACC_MODE	Mode d'accès invalide
-1073807339	VI_ERROR_TMO	Timeout expiré avant l'achèvement de l'opération
-1073807338	VI_ERROR_CLOSING_FAILED	Impossible de désallouer les structures de données précédemment allouées correspondant à cette session ou référence d'objet
-1073807332	VI_ERROR_INV_JOB_ID	Identificateur de tâche spécifié invalide
-1073807331	VI_ERROR_NSUP_ATTR	L'attribut spécifié n'est pas défini ou supporté par la ressource référencée
-1073807330	VI_ERROR_NSUP_ATTR_STATE	L'état spécifique de l'attribut n'est pas valide ou n'est pas supporté conformément à la définition de la ressource
-1073807329	VI_ERROR_ATTR_READONLY	L'attribut spécifié est en 'lecture seule'
-1073807322	VI_ERROR_INV_EVENT	Le type d'événement spécifié n'est pas supporté par la ressource

Tableau A-2. Codes d'erreur VISA (Suite)

Code d'erreur	Nom d'erreur	Description
-1073807321	VI_ERROR_INV_MECH	Mécanisme spécifié invalide
-1073807320	VI_ERROR_HNDLR_NINSTALLED	Un gestionnaire n'a pas été installé
-1073807319	VI_ERROR_INV_HNDLR_REF	La référence du gestionnaire donnée est invalide
-1073807318	VI_ERROR_INV_CONTEXT	Le contexte d'événement spécifié est invalide
-1073807308	VI_ERROR_RAW_WR_PROT_VIOL	Une violation du protocole de lecture brute s'est produite pendant le transfert
-1073807307	VI_ERROR_RAW_RD_PROT_VIOL	Une violation du protocole d'écriture brute s'est produite pendant le transfert
-1073807306	VI_ERROR_OUTP_PROT_VIOL	Le périphérique a rapporté une erreur de protocole de sortie pendant le transfert
-1073807305	VI_ERROR_INP_PROT_VIOL	Le périphérique a rapporté une erreur de protocole d'entrée pendant le transfert
-1073807304	VI_ERROR_BERR	Une erreur de bus s'est produite pendant le transfert
-1073807302	VI_ERROR_INV_SETUP	Impossible de démarrer l'opération parce que la configuration est invalide (attributs définis sur un état incohérent)
-1073807300	VI_ERROR_ALLOC	Ressources du système insuffisantes pour exécuter l'allocation de mémoire nécessaire
-1073807299	VI_ERROR_INV_MASK	Masque de buffer spécifié invalide
-1073807298	VI_ERROR_IO	Opération de lecture/écriture impossible à cause d'une erreur d'E/S
-1073807297	VI_ERROR_INV_FMT	Une spécification de format de la chaîne de caractères du format est invalide
-1073807295	VI_ERROR_NSUP_FMT	Une spécification de format de la chaîne de caractères du format n'est pas supportée
-1073807294	VI_ERROR_LINE_IN_USE	La ligne de déclenchement spécifiée est en cours d'utilisation
-1073807286	VI_ERROR_SRQ_NOCCURRED	Aucune requête de service n'a été reçue pour la session
-1073807282	VI_ERROR_INV_SPACE	Espace d'adresse spécifié invalide
-1073807279	VI_ERROR_INV_OFFSET	Offset spécifié invalide
-1073807276	VI_ERROR_NSUP_OFFSET	L'offset spécifié n'est pas accessible à partir de ce matériel
-1073807273	VI_ERROR_WINDOW_NMAPPED	La session spécifiée n'est pas actuellement mappée

Tableau A-2. Codes d'erreur VISA (Suite)

Code d'erreur	Nom d'erreur	Description
-1073807265	VI_ERROR_NLISTENERS	Condition d'absence d'auditeurs détectée (NRFD et NDAC sont désactivés)
-1073807264	VI_ERROR_NCIC	L'interface associée à cette session n'est actuellement pas le contrôleur en charge
-1073807257	VI_ERROR_NSUP_OPER	La référence de session ou d'objet donnée ne supporte pas cette opération
-1073807242	VI_ERROR_NSUP_WIDTH	La largeur spécifiée n'est pas supportée par ce matériel
-1073807239	VI_ERROR_INV_PROT	Le protocole spécifié est invalide
-1073807237	VI_ERROR_INV_SIZE	La taille de fenêtre spécifiée invalide
-1073807232	VI_ERROR_WINDOW_MAPPED	La session spécifiée contient déjà une fenêtre mappée
-1073807231	VI_ERROR_NIMPL_OPER	L'opération donnée n'est pas implémentée

Tableau A-3. Codes d'erreur d'analyse

Code d'erreur	Nom d'erreur	Description
0	NoErr	Aucune erreur ; l'appel a réussi
-20001	OutOfMemErr	Il ne reste pas suffisamment de mémoire pour exécuter la routine spécifiée
-20002	EqSamplesErr	Les séquences d'entrée doivent avoir la même taille
-20003	SamplesGTZeroErr	Le nombre d'échantillons doit être supérieur à zéro
-20004	SamplesGEZeroErr	Le nombre d'échantillons doit être supérieur ou égal à zéro
-20005	SamplesGEOneErr	Le nombre d'échantillons doit être supérieur ou égal à 1
-20006	SamplesGETwoErr	Le nombre d'échantillons doit être supérieur ou égal à 2
-20007	SamplesGEThreeErr	Le nombre d'échantillons doit être supérieur ou égal à 3
-20008	ArraySizeErr	Le nombre correct de valeurs de données ne figure pas dans les tableaux d'entrée de ce VI
-20009	PowerOfTwoErr	La taille de ce tableau d'entrée doit être une puissance de 2 : taille = 2^m , $0 < m < 23$
-20010	MaxXformSizeErr	La taille de transformation maximum a été dépassée
-20011	DutyCycleErr	Le facteur d'utilisation doit répondre à la condition : $0 \leq \text{facteur d'utilisation} \leq 100$

Tableau A-3. Codes d'erreur d'analyse (Suite)

Code d'erreur	Nom d'erreur	Description
-20012	CyclesErr	Le nombre de cycles doit être supérieur à zéro et inférieur ou égal au nombre d'échantillons
-20013	WidthLTSamplesErr	La largeur doit répondre à la condition : $0 < \text{largeur} < \text{échantillons}$
-20014	DelayWidthErr	Le retard doit répondre à la condition : $0 \leq (\text{retard} + \text{largeur}) < \text{échantillons}$
-20015	DtGEZeroErr	dt doit être supérieur ou égal à zéro
-20016	DtGTZeroErr	dt doit être supérieur à zéro
-20017	IndexLTSamplesErr	L'indice doit répondre à la condition : $0 \leq \text{indice} < \text{échantillons}$
-20018	IndexLengthErr	L'indice doit répondre à la condition : $0 \leq (\text{indice} + \text{longueur}) < \text{échantillons}$
-20019	UpperGELowerErr	La valeur haute doit être supérieure ou égale à la valeur basse
-20020	NyquistErr	La fréquence de coupure, f_c , doit répondre à la condition : $0 \leq f_c \leq \frac{f_s}{2}$
-20021	OrderGTZeroErr	L'ordre doit être supérieur à zéro
-20022	DecFactErr	Le facteur de décimation doit répondre à la condition : $0 < \text{décimation} \leq \text{échantillons}$
-20023	BandSpecErr	Les spécifications de bande doivent répondre à la condition : $0 \leq f_{basse} \leq f_{haute} \leq \frac{f_s}{2}$
-20024	RippleGTZeroErr	L'amplitude d'ondulation doit être supérieure à zéro
-20025	AttenGTZeroErr	L'atténuation doit être supérieure à zéro
-20026	WidthGTZeroErr	La largeur doit être supérieure à zéro
-20027	FinalGTZeroErr	La valeur finale doit être supérieure à zéro
-20028	AttenGTRippleErr	L'atténuation doit être supérieure à l'amplitude d'ondulation
-20029	StepSizeErr	La taille du pas, μ , doit répondre à la condition : $0 \leq \mu \leq 0.1$
-20030	LeakErr	Le coefficient de fuite doit répondre à la condition : $0 \leq \text{fuite} \leq \mu$

Tableau A-3. Codes d'erreur d'analyse (Suite)

Code d'erreur	Nom d'erreur	Description
-20031	EqRplDesignErr	Le filtre ne peut pas être conçu avec les valeurs d'entrée spécifiées
-20032	RankErr	Le rang du filtre doit répondre à la condition : $1 \leq (2 \times \text{rang} + 1) \leq \text{taille}$
-20033	EvenSizeErr	Le nombre de coefficients de ce filtre doit être impair
-20034	OddSizeErr	Le nombre de coefficients de ce filtre doit être pair
-20035	StdDevErr	L'écart-type doit être supérieur à zéro pour la normalisation
-20036	MixedSignErr	Les éléments du tableau des Valeurs Y doivent être non nuls ou bien tous positifs ou tous négatifs
-20037	SizeGTOOrderErr	Le nombre de points de données dans le tableau de Valeurs Y doit être supérieur à 2
-20038	IntervalsErr	Le nombre d'intervalles doit être supérieur à zéro
-20039	MatrixMulErr	Le nombre de colonnes de la première matrice n'est pas égal au nombre de rangs de la deuxième matrice ou vecteur
-20040	SquareMatrixErr	La matrice d'entrée doit être une matrice carrée
-20041	SingularMatrixErr	Le système d'équations ne peut pas être résolu parce que la matrice d'entrée est singulière
-20042	LevelsErr	Le nombre de niveaux est hors gamme
-20043	FactorErr	Le niveau des facteurs est hors gamme pour quelques données
-20044	ObservationsErr	Aucune observation n'a été faite à quelque niveau de facteur que ce soit
-20045	DataErr	Le nombre total de points de données doit être égal au produit des niveaux pour chaque facteur et des observations par cellule
-20046	OverflowErr	Il y a un débordement dans la valeur F calculée
-20047	BalanceErr	Les données sont déséquilibrées. Toutes les cellules doivent contenir le même nombre d'observations
-20048	ModelErr	Le modèle à effet aléatoire a été demandé alors que le modèle à effet fixe est requis
-20049	DistinctErr	Les valeurs x doivent être distinctes
-20050	PoleErr	La fonction d'interpolation a un pôle à la valeur requise

Tableau A-3. Codes d'erreur d'analyse (Suite)

Code d'erreur	Nom d'erreur	Description
-20051	ColumnErr	Toutes les valeurs de la première colonne de la matrice X doivent être 1
-20052	FreedomErr	Le nombre de degrés de liberté doit être supérieur ou égal à 1
-20053	ProbabilityErr	La probabilité doit être comprise entre zéro et 1
-20054	InvProbErr	La probabilité doit être supérieure ou égale à zéro et inférieure à 1
-20055	CategoryErr	Le nombre de catégories ou d'échantillons doit être supérieur à 1
-20056	TableErr	La table de contingence ne doit pas contenir de nombre négatif
-20061	InvSelectionErr	L'une des sélections d'entrée est invalide
-20062	MaxIterErr	Le nombre d'itérations maximum a été dépassé
-20063	PolyErr	Les coefficients de polynôme sont invalides
-20064	InitStateErr	Ce VI n'a pas été initialisé correctement
-20065	ZeroVectorErr	Le vecteur ne peut pas être nul

Tableau A-4. Codes d'erreur des VIs d'acquisition de données

Code d'erreur	Nom de l'erreur	Description
-10001	syntaxError	Une erreur a été détectée dans la chaîne de caractères d'entrée ; l'arrangement ou l'ordre des caractères dans la chaîne n'est pas cohérent avec l'ordre attendu
-10002	semanticsError	Une erreur a été détectée dans la chaîne de caractères d'entrée ; la syntaxe de la chaîne de caractères est correcte, mais certaines valeurs spécifiées ne sont pas cohérentes avec d'autres
-10003	invalidValueError	La valeur d'un paramètre numérique est invalide
-10004	valueConflictError	La valeur d'un paramètre numérique est incohérente avec un autre paramètre et la combinaison est en conséquence invalide
-10005	badDeviceError	Le paramètre de périphérique est invalide
-10006	badLineError	Le paramètre de ligne est invalide

Tableau A-4. Codes d'erreur des VIs d'acquisition de données (Suite)

Code d'erreur	Nom de l'erreur	Description
-10007	badChanError	Une voie est hors gamme pour le type de carte ou la configuration d'entrée, la combinaison de voies est interdite ou vous devez inverser l'ordre du balayage pour que la voie zéro soit la dernière
-10008	badGroupError	Le groupe est invalide
-10009	badCounterError	Le compteur est invalide
-10010	badCountError	Le compteur spécifié est trop petit ou trop grand ; ou le décompte de transfert d'E/S donné n'est pas approprié au buffer ou à la configuration de voie actuel
-10011	badIntervalError	La fréquence de balayage d'entrée analogique est trop rapide pour le nombre de voies et la fréquence d'horloge de voie ; ou la fréquence d'horloge de voie n'est pas supportée par la voie de compteur ou la voie d'E/S associée
-10012	badRangeError	La gamme de tension d'entrée ou de sortie analogique est invalide pour la voie spécifiée
-10013	badErrorCodeError	Le driver a retourné un code d'erreur non identifiable ou non listé
-10014	groupTooLargeError	La taille du groupe est trop grande pour la carte
-10015	badTimeLimitError	La limite temporelle est invalide
-10016	badReadCountError	Le nombre de lectures est invalide
-10017	badReadModeError	Le mode de lecture est invalide
-10018	badReadOffsetError	L'offset est illisible
-10019	badClkFrequencyError	La fréquence est invalide
-10020	badTimebaseError	La base de temps est invalide
-10021	badLimitsError	Les limites dépassent la gamme de la carte
-10022	badWriteCountError	Une mise à jour dans votre tableau de données est incomplète, ou vous tentez d'écrire au-delà de la fin du buffer interne, ou votre opération de sortie est continue et la longueur de votre tableau n'est pas un multiple de la moitié de la taille du buffer interne
-10023	badWriteModeError	Le mode d'écriture est hors de gamme ou est interdit
-10024	badWriteOffsetError	Ajouter l'offset d'écriture au repère d'écriture place ce dernier à l'extérieur du buffer interne

Tableau A-4. Codes d'erreur des VIs d'acquisition de données (Suite)

Code d'erreur	Nom de l'erreur	Description
-10025	limitsOutOfRangeError	Les limites d'entrée requises excèdent la capacité ou la configuration de la carte. Des limites alternatives ont été sélectionnées
-10026	badBufferSpecificationError	Le nombre de buffers requis ou la taille du buffer est interdit ; par exemple, la limite de buffer des Lab-PC est de 64 Kéchantillons, ou la carte ne supporte pas plusieurs buffers
-10027	badDAQEventError	Pour les DAQEvents 0 et 1, la valeur générale A doit être supérieure à 0 et inférieure à la taille du buffer interne. Si le DMA est utilisé pour le DAQEvent 1, la valeur générale A doit diviser parfaitement la taille du buffer interne. Si une TIO-10 est utilisée pour le DAQEvent 4, la valeur générale A doit être 1 ou 2
-10028	badFilterCutoffError	La fréquence de coupure spécifiée n'est pas valide pour ce périphérique
-10029	obsoleteFunctionError	La fonction que vous appelez n'est plus supportée par cette version du driver
-10030	badBaudRateError	Le débit en baud spécifié pour communiquer avec le port série n'est pas valide sur cette plate-forme
-10031	badChassisIDError	Le châssis SCXI spécifié ne correspond pas à un châssis SCXI configuré
-10032	badModuleSlotError	L'emplacement du module SCXI spécifié est invalide ou correspond à un emplacement vide
-10033	invalidWinHandleError	Le handle de la fenêtre communiqué à la fonction est invalide
-10034	noSuchMessageError	Aucun message configuré ne correspond à celui que vous avez essayé d'effacer
-10080	badGainError	Le gain est invalide
-10081	badPretrigCountError	Le nombre d'échantillons de pré-déclenchement est invalide
-10082	badPosttrigCountError	Le nombre d'échantillons de post-déclenchement est invalide
-10083	badTrigModeError	Le mode de déclenchement est invalide
-10084	badTrigCountError	Le nombre de déclenchements est invalide
-10085	badTrigRangeError	La gamme de déclenchement ou la fenêtre d'hystérésis du déclenchement est invalide
-10086	badExtRefError	La valeur de référence externe est invalide

Tableau A-4. Codes d'erreur des VIs d'acquisition de données (Suite)

Code d'erreur	Nom de l'erreur	Description
-10087	badTrigTypeError	Le paramètre du type de déclenchement est invalide
-10088	badTrigLevelError	Le niveau du déclenchement est invalide
-10089	badTotalCountError	Le nombre total n'est pas cohérent avec la taille du buffer et le nombre de balayages de pré-déclenchement ou avec le type de carte
-10090	badRPGError	Les paramètres de gamme, de polarité et de gain individuels sont valides mais la combinaison spécifiée n'est pas autorisée
-10091	badIterationsError	Vous avez tenté d'utiliser une valeur invalide pour le paramètre des itérations. La valeur des itérations doit être supérieure ou égale à zéro. Votre périphérique est peut-être limité à deux valeurs uniquement, 0 et 1
-10092	lowScanIntervalError	Certains périphériques requièrent un intervalle de temps entre le dernier échantillon d'un balayage et le premier du balayage suivant. L'intervalle de balayage que vous avez spécifié ne procure pas un intervalle suffisamment grand pour la carte. Reportez-vous à la fonction <code>SCAN_Start</code> dans l'interface de langage API pour une explication plus détaillée
-10093	fifoModeError	La génération de forme d'onde en mode FIFO ne peut pas être utilisée parce qu'une condition au moins n'est pas remplie
-10100	badPortWidthError	La largeur du port numérique requise n'est pas un multiple de la largeur de port du matériel ou ne peut pas être atteinte par le matériel DAQ
-10120	gpctrBadApplicationError	Application utilisée invalide
-10121	gpctrBadCtrNumberError	Numéro de compteur utilisé invalide
-10122	gpctrBadParamValueError	Valeur de paramètre utilisée invalide
-10123	gpctrBadParamIDError	Paramètre ID utilisé invalide
-10124	gpctrBadEntityIDError	Entité ID utilisée invalide
-10125	gpctrBadActionError	Action utilisée invalide
-10200	EEPROMreadError	Impossible de lire des données de l'EEPROM

Tableau A-4. Codes d'erreur des VIs d'acquisition de données (Suite)

Code d'erreur	Nom de l'erreur	Description
-10201	EEPROMwriteError	Impossible d'écrire des données dans l'EEPROM
-10240	noDriverError	L'interface du driver n'a pas pu localiser ou ouvrir le driver
-10241	oldDriverError	L'un des fichiers du driver ou l'utilitaire de configuration n'est pas applicable
-10242	functionNotFoundError	La fonction spécifiée ne se trouve pas dans le driver
-10244	deviceInitError	Une erreur d'initialisation matérielle s'est produite lorsque le driver a tenté de configurer le périphérique spécifié
-10245	osInitError	Une erreur du système d'exploitation s'est produite lorsque le driver a tenté d'exécuter une opération, ou le système d'exploitation ne supporte pas une opération exécutée par le driver
-10246	communicationsError	Le driver est incapable de communiquer avec le périphérique externe spécifié
-10248	dupAddressError	Les adresses de base de deux ou plusieurs périphériques sont identiques ; par conséquent, le driver est incapable d'accéder au périphérique spécifié
-10249	intConfigError	La configuration d'interruption est incorrecte étant donné les capacités de l'ordinateur ou du périphérique
-10250	dupIntError	Les niveaux d'interruption d'un ou de plusieurs périphériques sont identiques
-10251	dmaConfigError	La configuration DMA est incorrecte étant donné les capacités de l'ordinateur, du contrôleur ou du périphérique DMA
-10252	dupDMAError	Les voies DMA de deux ou de plusieurs périphériques sont identiques
-10253	jumperlessBoardError	Impossible de trouver une ou plusieurs cartes sans cavalier que vous avez configurées à l'aide de l'utilitaire de configuration NI-DAQ

Tableau A-4. Codes d'erreur des VIs d'acquisition de données (Suite)

Code d'erreur	Nom de l'erreur	Description
-10254	DAQCardConfError	Impossible de configurer la carte DAQ pour l'une des raisons suivantes : <ol style="list-style-type: none"> 1. La version correcte de la carte et du logiciel gérant les emplacements de carte n'est pas installée. 2. La carte se trouvant dans l'emplacement PCMCIA n'est pas une carte DAQ. 3. L'adresse de base et/ou le niveau d'interruption requis ne sont pas disponibles selon le gestionnaire de ressources de la carte et des emplacements de cartes. 4. Les services de cartes n'ont pas réussi à se charger à cause d'une mémoire disponible inférieure à 1 méga-octet sous Windows 3.1. Essayez différentes configurations ou utilisez AutoAssign dans l'utilitaire de configuration NI-DAQ. Une mémoire inférieure à 1 méga-octet doit être disponible pour configurer la carte DAQ dans Win 3.x.
-10255	remoteChassisDriverInitError	Une erreur s'est produite lors de l'initialisation du driver pour l'unité 'remote SCXI'
-10256	comPortOpenError	Une erreur s'est produite lors de l'ouverture du port COM spécifié
-10257	baseAddressError	L'adresse de base spécifiée dans l'utilitaire de configuration est incorrecte
-10258	dmaChannel1Error	La voie DMA 1 spécifiée dans l'utilitaire de configuration ou par le système d'exploitation est incorrect
-10259	dmaChannel2Error	La voie DMA 2 spécifiée dans l'utilitaire de configuration ou par le système d'exploitation est incorrecte
-10260	dmaChannel3Error	La voie DMA 3 spécifiée dans l'utilitaire de configuration ou par le système d'exploitation est incorrecte
-10261	userModeToKernelModeCallError	Le code du mode utilisateur a échoué lors de l'appel du mode kernel
-10340	noConnectError	Aucun signal/ligne RTSI n'est connecté ou le signal et la ligne spécifiés ne sont pas connectés
-10341	badConnectError	Le signal/ligne RTSI ne peut pas être connecté comme spécifié

Tableau A-4. Codes d'erreur des VIs d'acquisition de données (Suite)

Code d'erreur	Nom de l'erreur	Description
-10342	multConnectError	Le signal RTSI spécifié est déjà transporté par une ligne RTSI ou la ligne RTSI spécifiée est déjà transportée par un signal RTSI
-10343	SCXIConfigError	Les paramètres de configuration SCXI spécifiés sont invalides ou la fonction ne peut pas être exécutée conformément à la configuration SCXI courante
-10344	chassisSynchedError	L'unité 'Remote SCXI' n'est pas synchronisée avec l'hôte. Réinitialisez encore le châssis pour le synchroniser avec l'hôte
-10345	chassisMemAllocError	La mémoire demandée ne peut pas être attribuée à l'unité 'Remote SCXI' pour l'opération spécifiée
-10346	badPacketError	Le paquet reçu par l'unité 'Remote SCXI' est invalide. Vérifiez les connexions du câble port série
-10347	chassisCommunicationError	Une erreur s'est produite lors de l'envoi d'un paquet au châssis déporté. Vérifiez les connexions du câble port série
-10348	waitingForReprogError	L'unité 'Remote SCXI' est en mode de re-programmation et attend les commandes de re-programmation de l'hôte (Utilitaire de configuration NI-DAQ)
-10349	SCXIModuleTypeConflictError	L'ID de module lue dans le module SCXI ne correspond pas au type de module configuré
-10370	badScanListError	La liste des balayages est invalide ; par exemple, vous mélangez les voies AMUX-64T, les voies intégrées et les voies SCXI désactivées, ou vous avez spécifié une voie de départ différente pour le même module SCXI. Ou bien, le driver tente de définir des distributions du gain complexes via des voies SCXI du même module en manipulant la liste des balayages et retourne cette erreur en cas d'échec
-10400	userOwnedRsrcError	La ressource spécifiée appartient à l'utilisateur et le driver ne peut pas y accéder ni la modifier
-10401	unknownDeviceError	Le périphérique spécifié n'est pas un produit de National Instruments, ou le driver ne supporte pas le périphérique (par exemple, le driver a été mis sur le marché avant que le périphérique soit supporté.)
-10402	deviceNotFoundError	Il n'y pas de périphérique dans l'emplacement ou à l'adresse spécifiés
-10404	noLineAvailError	Aucune ligne n'est disponible
-10405	noChanAvailError	Aucune voie n'est disponible

Tableau A-4. Codes d'erreur des VIs d'acquisition de données (Suite)

Code d'erreur	Nom de l'erreur	Description
-10406	noGroupAvailError	Aucun groupe n'est disponible
-10407	lineBusyError	La ligne spécifiée est en cours d'utilisation
-10408	chanBusyError	La voie spécifiée est en cours d'utilisation
-10409	groupBusyError	Le groupe spécifié est en cours d'utilisation
-10410	relatedLCGBusyError	Une ligne, une voie ou un groupe dédié(e) est en cours d'utilisation ; si le driver configure la ligne, la voie ou le groupe spécifié(e), les lignes de configuration, de données ou de handshaking de la ligne, de la voie ou du groupe dédié(e) seront affectées
-10411	counterBusyError	Le compteur spécifié est en cours d'utilisation
-10412	noGroupAssignError	Aucun groupe n'est assigné ou la ligne ou la voie spécifiée ne peut pas être assignée à un groupe
-10413	groupAssignError	Un groupe est déjà assigné ou la ligne ou la voie est déjà assignée à un groupe
-10414	reservedPinError	Le signal sélectionné exige qu'une broche soit réservée et configurée uniquement par NI-DAQ. Vous ne pouvez pas configurer cette broche vous-même
-10415	externalMuxSupportError	Cette fonction ne supporte pas ce périphérique lorsqu'il est connecté à un multiplexeur externe (tel qu'un AMUX-64T ou SCXI)
-10440	sysOwnedRsrcError	La ressource spécifiée appartient au driver et l'utilisateur ne peut pas y accéder ni la modifier
-10441	memConfigError	Aucune mémoire n'est configurée pour supporter le mode de transfert de données ou la mémoire configurée ne supporte pas le mode de transfert de données courant. (Si les transferts de blocs sont en cours d'utilisation, la mémoire doit être capable d'effectuer des transferts de blocs.)
-10442	memDisabledError	La mémoire spécifiée est désactivée ou indisponible conformément au mode d'adressage courant
-10443	memAlignmentError	Le buffer de transfert n'est pas correctement aligné pour le mode de transfert de données courant. Par exemple, le buffer n'est pas à la bonne adresse ; il n'est pas aligné sur une limite de 32 bits, sur une limite de 512 bits et ainsi de suite. Ou bien, le driver ne peut pas aligner le buffer parce que ce dernier est trop petit

Tableau A-4. Codes d'erreur des VIs d'acquisition de données (Suite)

Code d'erreur	Nom de l'erreur	Description
-10445	memLockError	Le buffer de transfert ne peut pas être verrouillé dans la mémoire physique. Sur des machines PC AT, des portions du buffer d'acquisition de données utilisé par les DMA se trouvent peut-être dans une zone DMA, comme par exemple, au-dessus de 16 méga-octets
-10446	memPageError	Le buffer de transfert contient un saut de page ; en cas de saut de page, les ressources du système peuvent exiger une re-programmation
-10447	memPageLockError	L'environnement d'exploitation ne peut pas accorder de verrouillage de page
-10448	stackMemError	Le driver ne peut pas continuer d'analyser une entrée de chaîne de caractères à cause des limitations de pile
-10449	cacheMemError	Une erreur relative à la mémoire cache s'est produite ou la mise en mémoire cache n'est pas supportée dans le mode courant
-10450	physicalMemError	Une erreur matérielle s'est produite dans la mémoire physique ou il n'y a pas de mémoire à l'adresse spécifiée
-10451	virtualMemError	Le driver est incapable de faire un transfert de buffer adjacent dans la mémoire virtuelle et ne peut donc pas verrouiller le buffer dans la mémoire physique ; par conséquent, vous ne pouvez pas utiliser le buffer pour des transferts de DMA
-10452	noIntAvailError	Aucun niveau d'interruption n'est disponible
-10453	intInUseError	Le niveau d'interruption spécifié est déjà utilisé par un autre périphérique
-10454	noDMACError	Aucun contrôleur DMA n'est disponible dans le système
-10455	noDMAAvailError	Aucune voie DMA n'est disponible
-10456	DMAInUseError	La voie DMA spécifiée est déjà utilisée par un autre périphérique
-10457	badDMAGroupError	Le mode DMA ne peut pas être configuré dans le groupe spécifié parce qu'il est trop petit, trop grand ou mal aligné. Consultez le manuel de l'utilisateur du périphérique en question pour déterminer les ramifications du groupe conformément aux DMA
-10458	diskFullError	Un débordement de disque s'est produit lors de la tentative d'écriture vers un fichier

Tableau A-4. Codes d'erreur des VIs d'acquisition de données (Suite)

Code d'erreur	Nom de l'erreur	Description
-10459	DLLInterfaceError	La DLL n'a pas pu être appelée à cause d'une erreur d'interface
-10460	interfaceInteractionError	Vous avez mélangé des VIs de la bibliothèque DAQ et de la bibliothèque de compatibilité DAQ_(VIs de LabVIEW 2.2). Vous ne pouvez utiliser les deux bibliothèques qu'en exécutant le VI DAQ "Réinitialiser un périphérique" (Device Reset.vi) avant d'appeler les VIs de compatibilité ou en exécutant le VI "Réinitialiser une carte" (Board Reset.vi) avant d'appeler les VI DAQ
-10480	muxMemFullError	La liste des balayages est trop longue pour la mémoire de gain du multiplexeur de la carte
-10481	bufferNotInterleavedError	Vous devez fournir un buffer simple ou des données entrelacées et les voies doivent être en ordre croissant. Vous ne pouvez pas utiliser de DMA pour transférer des données provenant de deux buffers ; vous pouvez cependant utiliser des interruptions
-10540	SCXIModuleNotSupportedError	Au moins un des modules SCXI spécifiés n'est pas supporté par l'opération
-10541	TRIG1ResourceConflict	CTRB1 pilote COUTB1. Cependant, CTRB1 pilote également TRIG1. Ce conflit peut entraîner des résultats imprévisibles lors du balayage du châssis
-10600	noSetupError	Aucune opération de configuration n'a été effectuée pour les ressources spécifiées. Ou, certaines ressources exigent que les appels soient faits dans un ordre spécifié pour une configuration correcte
-10601	multSetupError	Les ressources spécifiées ont déjà été configurées par une opération de configuration
-10602	noWriteError	Aucune donnée de sortie n'a été écrite dans le buffer de transfert
-10603	groupWriteError	Les données de sortie relatives à un groupe sont destinées à une seule voie ou à des voies consécutives
-10604	activeWriteError	Une fois que la génération de données a démarré, seuls les buffers de transfert dans lesquels des données ont été initialement écrites peuvent être mis à jour. Si le mode DMA est actif et qu'un buffer de transfert unique contient des données entrelacées par voies, de nouvelles données doivent être fournies à toutes les voies de sortie utilisant actuellement la voie DMA
-10605	endWriteError	Aucune donnée n'a été écrite dans le buffer de transfert parce que le bloc de données final a déjà été chargé

Tableau A-4. Codes d'erreur des VIs d'acquisition de données (Suite)

Code d'erreur	Nom de l'erreur	Description
-10606	notArmedError	La ressource spécifiée n'est pas activée
-10607	armedError	La ressource spécifiée est déjà activée
-10608	noTransferInProgError	Aucun transfert n'est en cours pour la ressource spécifiée
-10609	transferInProgError	Un transfert est déjà en cours pour la ressource spécifiée ou l'opération n'est pas autorisée parce que le périphérique est en train d'effectuer des transferts, probablement avec des ressources différentes
-10610	transferPauseError	Une voie de sortie unique dans un groupe ne peut pas être interrompue si les données de sortie du groupe sont entrelacées
-10611	badDirOnSomeLinesError	Certaines lignes de la voie spécifiée ne sont pas configurées pour la direction de transfert spécifiée. Pour un transfert d'écriture, certaines lignes ont été configurées en entrée. Pour un transfert de lecture, certaines lignes ont été configurées en sortie
-10612	badLineDirError	La ligne spécifiée ne supporte pas la direction de transfert spécifiée
-10613	badChanDirError	La voie spécifiée ne supporte pas la direction de transfert spécifiée
-10614	badGroupDirError	Le groupe spécifié ne supporte pas la direction de transfert spécifiée
-10615	masterClkError	La configuration de l'horloge maître est invalide
-10616	slaveClkError	La configuration de l'horloge esclave est invalide
-10617	noClkSrcError	Aucun signal source n'a été assigné à la ressource de l'horloge
-10618	badClkSrcError	Le signal source spécifié ne peut pas être assigné à la ressource de l'horloge
-10619	multClkSrcError	Un signal source a déjà été assigné à la ressource de l'horloge
-10620	noTrigError	Aucun signal de déclenchement n'a été assigné à la ressource du déclenchement
-10621	badTrigError	Le signal de déclenchement spécifié ne peut pas être assigné à la ressource du déclenchement
-10622	preTrigError	Le mode de pré-déclenchement n'est pas supporté ou n'est pas disponible dans la configuration actuelle ou aucune source de pré-déclenchement n'a été assignée

Tableau A-4. Codes d'erreur des VIs d'acquisition de données (Suite)

Code d'erreur	Nom de l'erreur	Description
-10623	postTrigError	Aucune source de post-déclenchement n'a été assignée
-10624	delayTrigError	Le mode de déclenchement retardé n'est pas supporté ou n'est pas disponible dans la configuration actuelle ou aucune source de retard n'a été assignée
-10625	masterTrigError	La configuration du déclenchement maître est invalide
-10626	slaveTrigError	La configuration du déclenchement esclave est invalide
-10627	noTrigDrvError	Aucun signal n'a été assigné à la ressource du déclenchement
-10628	multTrigDrvError	Un signal a déjà été assigné à la ressource du déclenchement
-10629	invalidOpModeError	Le mode de fonctionnement spécifié est invalide ou les ressources n'ont pas été configurées pour le mode de fonctionnement spécifié
-10630	invalidReadError	Les paramètres spécifiés pour lire des données sont invalides dans le contexte de l'acquisition. Par exemple, une tentative de lecture de 0 octet du buffer de transfert ou une tentative de lire au-delà de la fin du buffer de transfert a été faite
-10631	noInfiniteModeError	Les transferts d'entrée et de sortie en continu ne sont pas autorisés dans le mode de fonctionnement courant
-10632	someInputsIgnoredError	Certaines entrées ont été ignorées parce qu'elles ne sont pas associées au mode de fonctionnement courant
-10633	invalidRegenModeError	Le mode de régénération de sortie analogique n'est pas autorisé sur cette carte
-10634	noContTransferInProgressError	Aucun transfert (double buffer) en continu n'est en cours pour la ressource spécifiée
-10635	invalidSCXIOpModeError	Le mode de fonctionnement SCXI spécifié dans un appel de configuration est invalide ; ou un module n'est pas dans le mode de fonctionnement lui permettant d'effectuer l'appel de la fonction
-10636	noContWithSynchError	Vous ne pouvez pas démarrer une opération (double buffer) de façon continue avec un appel de fonction synchrone
-10637	bufferAlreadyConfigError	Tentative de configuration d'un buffer après que ce dernier a déjà été configuré. Vous ne pouvez configurer un buffer qu'une seule fois
-10680	badChanGainError	Toutes les voies de cette carte doivent avoir le même gain

Tableau A-4. Codes d'erreur des VIs d'acquisition de données (Suite)

Code d'erreur	Nom de l'erreur	Description
-10681	badChanRangeError	Toutes les voies de cette carte doivent avoir la même gamme
-10682	badChanPolarityError	Toutes les voies de cette carte doivent avoir la même polarité
-10683	badChanCouplingError	Toutes les voies de cette carte doivent avoir le même couplage
-10684	badChanInputModeError	Toutes les voies de cette carte doivent avoir le même mode d'entrée
-10685	clkExceedsBrdsMaxConvRateError	La fréquence d'horloge sélectionnée dépasse la fréquence maximum recommandée sur cette carte
-10686	scanListInvalidError	Une modification de la configuration a invalidé la liste des balayages
-10687	bufferInvalidError	Une modification de la configuration a invalidé le buffer d'acquisition ou un buffer d'acquisition n'a pas été configuré
-10688	noTrigEnabledError	Le nombre total de balayages et de balayages de pré-déclenchement impliquent qu'un démarrage de déclenchement est prévu mais aucun déclenchement n'a été activé
-10689	digitalTrigBError	Le déclenchement numérique B est invalide pour le nombre total de balayages et les balayages de pré-déclenchement spécifiés
-10690	digitalTrigAandBError	Cette carte ne permet pas que les déclenchements numériques A et B soient activés en même temps
-10691	extConvRestrictionError	Une horloge d'échantillonnage externe combinée avec une horloge de balayage externe, un déclenchement de démarrage ou un déclenchement d'arrêt n'est pas autorisée sur cette carte
-10692	chanClockDisabledError	L'acquisition ne peut pas être démarrée parce que l'horloge est désactivée
-10693	extScanClockError	Une horloge de balayage externe ne peut pas être utilisée lors d'un balayage unique d'une voie unique
-10694	unsafeSamplingFreqError	La fréquence d'échantillonnage dépasse la fréquence limite maximum du matériel, des gains et des filtres utilisés

Tableau A-4. Codes d'erreur des VIs d'acquisition de données (Suite)

Code d'erreur	Nom de l'erreur	Description
-10695	DMANotAllowedError	Vous avez configuré une opération qui requiert l'utilisation d'interruptions. Le mode DMA n'est pas autorisé. Par exemple, certains événements DAQ, tels que l'envoi de messages et les occurrences de LabVIEW, requièrent des interruptions
-10696	multiRateModeError	Le balayage à fréquence multiple ne peut pas être utilisé avec des systèmes AMUX-64, SCXI ou des acquisitions pré-déclenchées
-10697	rateNotSupportedError	NI-DAQ n'a pas pu convertir l'association base de temps/intervalle pour qu'il soit conforme aux capacités matérielles courantes de la carte spécifiée
-10698	timebaseConflictError	Vous ne pouvez pas utiliser cette combinaison de bases de temps du balayage et de l'horloge d'échantillonnage pour la carte spécifiée
-10699	polarityConflictError	Vous ne pouvez pas utiliser cette combinaison de polarités de source d'horloge d'échantillonnage et de balayage pour cette opération et cette carte
-10700	signalConflictError	Vous ne pouvez pas utiliser cette combinaison de sources de signal d'horloge de conversion et de balayage pour cette opération et cette carte
-10701	noLaterUpdateError	L'appel est resté sans effet parce que la voie spécifiée n'a pas été définie pour des mises à jour internes ultérieures
-10702	prePostTriggerError	Le pré-déclenchement et le post-déclenchement ne peuvent pas être utilisés simultanément sur les périphériques Lab et série 1200
-10710	noHandshakeModeError	Le port spécifié n'a pas été configuré pour le handshaking
-10720	noEventCtrError	Le compteur spécifié n'est pas configuré pour des opérations de comptage d'événements
-10740	SCXITrackHoldError	Un signal a déjà été assigné à la ligne de déclenchement de l'échantillonneur-bloqueur SCXI ou un appel de contrôle n'est pas approprié parce que le module spécifié n'est pas configuré pour une opération à une voie
-10780	sc2040InputModeError	Lorsqu'un SC-2400 est connecté à votre périphérique, toutes les voies d'entrées analogiques doivent être configurées pour un mode d'entrée différentiel
-10781	outputTypeMustBeVoltageError	La polarité de la voie de sortie ne peut pas être bipolaire lors de la génération de courants

Tableau A-4. Codes d'erreur des VIs d'acquisition de données (Suite)

Code d'erreur	Nom de l'erreur	Description
-10782	sc2040HoldModeError	L'opération spécifiée ne peut pas être effectuée avec le SC-2400 configuré en mode de rétention
-10783	calConstPolarityConflictError	Les constantes d'étalonnage dans la zone de chargement ont une polarité différente de celle de la configuration courante. Vous devez donc charger les constantes d'usine
-10800	timeOutError	L'opération n'a pas pu être terminée dans la limite temporelle
-10801	calibrationError	Une erreur s'est produite pendant le processus d'étalonnage
-10802	dataNotAvailError	Le volume de données requis n'a pas encore été acquis
-10803	transferStoppedError	Le transfert a été arrêté pour éviter la régénération de données de sortie
-10804	earlyStopError	Le transfert s'est arrêté avant d'atteindre la fin du buffer de transfert
-10805	overRunError	La source d'horloge de tâche d'entrée est plus rapide que la fréquence d'horloge maximum supportée par le périphérique. Si vous permettez au driver de calculer la fréquence d'horloge de la voie d'entrée analogique, le driver la définit en fonction du type de carte ; vous devez donc vérifier que le type de carte est correct dans l'utilitaire de configuration.
-10806	noTrigFoundError	Aucune valeur de déclenchement n'a été trouvée dans le buffer de transfert d'entrée
-10807	earlyTrigError	Le déclenchement s'est produit avant l'acquisition d'un nombre suffisant de données de pré-déclenchement
-10808	LPTCommunicationError	Une erreur s'est produite lors de la communication du port parallèle avec le périphérique DAQ
-10809	gateSignalError	Tentative de démarrage d'une mesure de largeur d'une impulsion en phase de mesure (par exemple, phase haute pour fenêtrage sur niveau haut)
-10840	internalDriverError	Une erreur inattendue s'est produite dans le driver lors de l'exécution de l'opération donnée
-10841	firmwareError	Le microprogramme ne supporte pas l'opération spécifiée ou l'opération microprogramme n'a pas pu se terminer à cause d'un problème d'intégrité des données
-10842	hardwareError	Le matériel ne répond pas à l'opération spécifiée ou la réponse du matériel n'est pas compatible avec la fonctionnalité du matériel

Tableau A-4. Codes d'erreur des VIs d'acquisition de données (Suite)

Code d'erreur	Nom de l'erreur	Description
-10843	underFlowError	A cause des limitations du système, le driver n'a pas pu écrire des données dans le périphérique de façon suffisamment rapide pour suivre le débit du périphérique
-10844	underWriteError	Les nouvelles données n'ont pas été écrites dans le buffer de transfert de sortie avant que le driver ait tenté de transférer les données vers le périphérique
-10845	overflowError	A cause des limitations du système, le driver n'a pas pu lire les données du périphérique de façon suffisamment rapide pour suivre le débit du périphérique ; la mémoire du périphérique intégrée a rapporté une erreur de débordement
-10846	overwriteError	Le driver a écrit de nouvelles données dans le buffer de transfert d'entrée avant que les données précédemment acquises aient été lues
-10847	dmaChainingError	Les informations sur le nouveau buffer n'étaient pas disponibles au moment de l'interruption de chaînage DMA ; les transferts DMA se termineront à la fin du buffer de transfert actuellement actif
-10848	noDMACountAvailError	Le driver n'a pas pu obtenir des données valides du registre de comptage du transfert dans le contrôleur DMA
-10849	openFileError	Le fichier de configuration n'a pas pu être ouvert
-10850	closeFileError	Impossible de fermer un fichier
-10851	fileSeekError	Impossible d'effectuer une recherche dans un fichier
-10852	readFileError	Impossible de lire dans un fichier
-10853	writeFileError	Impossible d'écrire dans un fichier
-10854	miscFileError	Une erreur s'est produite lors de l'accès à un fichier
-10855	osUnsupportedError	NI-DAQ ne supporte pas l'opération courante sur cette version particulière du système d'exploitation
-10856	osError	Une erreur inattendue s'est produite dans le système d'exploitation lors de l'exécution de l'opération donnée
-10857	internalKernelError	Une erreur inattendue s'est produite dans le kernel lors de l'exécution de l'opération donnée

Tableau A-4. Codes d'erreur des VIs d'acquisition de données (Suite)

Code d'erreur	Nom de l'erreur	Description
-10880	updateRateChangeError	<p>Une modification de la fréquence de mise à jour n'est pas possible pour le moment pour l'une des raisons suivantes :</p> <ol style="list-style-type: none"> 1. Lorsque la génération de la forme d'onde est en progression, vous ne pouvez pas modifier l'intervalle de base de temps. 2. Lorsque vous effectuez des modifications successives, vous devez laisser à chaque modification suffisamment de temps pour entrer en vigueur avant de demander d'autres modifications
-10881	partialTransferCompleteError	Vous ne pouvez pas faire d'autres transferts après un transfert partiel réussi
-10882	daqPollDataLossError	Les données collectées sur l'unité 'remote SCXI' ont été écrasées avant d'avoir pu être transférées dans le buffer de l'hôte. Essayez, si possible, d'utiliser une fréquence d'acquisition des données plus lente
-10883	wfmPollDataLossError	De nouvelles données n'ont pas pu être transférées dans le buffer de formes d'ondes de l'unité 'remote SCXI' suffisamment rapidement pour suivre la fréquence de mise à jour de la forme d'onde. Essayez, si possible, d'utiliser une fréquence de mise à jour plus lente
-10884	pretrigReorderError	Les données n'ont pas pu être réorganisées après la fin d'une acquisition à pré-déclenchement
-10920	gpctrDataLossError	Un ou plusieurs points de données ont peut-être été perdus au cours des opérations GPCTR bufférisées à cause des limites de vitesse de votre système
-10940	chassisResponseTimeoutError	L'unité 'remote SCXI' n'a pas envoyé de réponse dans la limite temporelle spécifiée
-10941	reprogrammingFailedError	Echec de la reprogrammation de l'unité 'remote SCXI'. Veuillez recommencer
-10942	invalidResetSignatureError	L'hôte a envoyé une signature de remise à zéro à l'unité 'remote SCXI'
-10943	chassisLockupError	La routine de service d'interruption de l'unité 'remote SCXI' prend plus de temps que nécessaire. Vous n'avez pas besoin de remettre à zéro votre unité 'remote SCXI' ; toutefois, vous devez réinitialiser et redémarrer votre acquisition de données

Tableau A-5. Codes d'erreur AppleEvents

Code d'erreur	Nom de l'erreur	Description
-1700	errAECOercionFail	Des données n'ont pas pu être contraintes au type de descripteur requis
-1701	errAEDescNotFound	L'enregistrement du descripteur n'a pas été trouvé
-1702	errAECorruptData	Les données d'un AppleEvent n'ont pas pu être lues
-1703	errAEWrongDataType	Type de descripteur incorrect
-1704	errAENotAEDesc	Enregistrement du descripteur invalide
-1705	errAEBadListItem	L'opération impliquant un élément de la liste a échoué
-1706	errAENewerVersion	Une version récente du Gestionnaire d'AppleEvent est nécessaire
-1707	errAENotAppleEvent	L'événement n'est pas un AppleEvent
-1708	errAERReplyNotValid	AEResetTimer a reçu un paramètre de réponse invalide
-1709	errAERERReplyNotValid	Un paramètre de réponse invalide a été communiqué à AEResetTimer
-1710	errAEUnknownSendMode	Un mode d'envoi invalide a été communiqué
-1711	errAEWaitcanceled	Utilisateur a supprimé des boucles d'attente pour les réponses ou les réceptions
-1712	errAETimeout	AppleEvent différé
-1713	errAENoUserInteraction	Aucun dialogue avec l'utilisateur n'est permis
-1714	errAENotASpecialFunction	Mot clé incorrect pour une fonction particulière
-1715	errAEPParamMissed	Le gestionnaire n'a pas reçu tous les paramètres requis
-1716	errAEUnknownAddressType	Type d'adresse d'AppleEvent inconnu
-1717	errAEHandlerNotFound	Aucun gestionnaire dans les tableaux de distribution ne correspond aux paramètres de l'AEGetEventHandler ou de l'AEGetCoercionHandler
-1718	errAERReplyNotArrived	Le contenu de la réponse à laquelle vous accédez n'est pas encore arrivé
-1719	errAEIllegalIndex	L'indice est en dépassement de gamme lors d'une opération d'écriture

Tableau A-6. Codes d'erreur des drivers d'instruments

Etat	Description de l'état	Défini par
0	Aucune erreur ; l'appel a réussi	—
-1200	Syntaxe de chaîne de caractères invalide	Bibliothèque de transition VISA
-1201	Erreur lors de la recherche de l'instrument	Bibliothèque de transition VISA
-1202	Impossible d'initialiser l'interface ou l'instrument	Bibliothèque de transition VISA
-1205	Gestionnaire d'instrument invalide	Bibliothèque de transition VISA
-1210	Paramètre hors gamme	VI de driver d'instrument
-1215	Erreur lors de la fermeture de l'instrument	Bibliothèque de transition VISA
-1218	Erreur lors de l'obtention de l'attribut	Bibliothèque de transition VISA
-1219	Erreur lors de la définition des attributs	Bibliothèque de transition VISA
-1223	L'interrogation pour l'identification de l'instrument a échoué	VI d'initialisation de driver d'instrument
-1224	Erreur lors de la réinitialisation de l'instrument	Bibliothèque de transition VISA
-1225	Erreur dans le déclenchement de l'instrument	Bibliothèque de transition VISA
-1226	Erreur lors de l'appel de l'instrument	Bibliothèque de transition VISA
-1230	Erreur lors de l'écriture dans l'instrument	Bibliothèque de transition VISA
-1231	Erreur lors de la lecture à partir de l'instrument	Bibliothèque de transition VISA
-1236	Erreur lors de l'interprétation de la réponse de l'instrument	VI de driver d'instrument
-1240	Instrument différé	Bibliothèque de transition VISA
-1250	Erreur lors du mappage de l'adresse VXI	Bibliothèque de transition VISA
-1251	Erreur lors du démappage de l'adresse VXI	Bibliothèque de transition VISA
-1252	Erreur lors de l'accès à l'adresse VXI	Bibliothèque de transition VISA
-1260	Fonction non supportée par le contrôleur	Bibliothèque de transition VISA
-1300	Erreur spécifique à l'instrument	—
-13xx	Codes d'erreur spécifiés par le réalisateur	—

Tableau A-7. Codes d'erreur PPC

Code	Nom	Description
-900	notInitErr	PPCToolbox n'a pas été initialisée
-902	nameTypeErr	LocationKindSelector invalide ou impropre dans LocationName
-903	noPortErr	Nom de port invalide. Impossible d'ouvrir le port ou port RefNum incorrect
-904	noGlobalsErr	Le système est incapable d'allouer de la mémoire. C'est une erreur critique et vous devez redémarrer
-905	localOnlyErr	L'activité du réseau est actuellement désactivée
-906	destPortErr	Il n'a pas de port à la destination
-907	sessTableErr	PPCToolbox est incapable de créer une session
-908	noSessionErr	Numéro de référence de la session invalide
-909	badReqErr	Paramètre incorrect ou état invalide pour cette opération
-910	portNameExistsErr	Un autre port portant ce nom est déjà ouvert (peut-être dans une autre application)
-911	noUserNameErr	Nom de l'utilisateur inconnu sur la machine de destination
-912	userRejectErr	La destination a rejeté la demande de la session
-913	noMachineNameErr	L'utilisateur n'a pas nommé son Macintosh dans le panneau de contrôle de configuration du réseau
-914	noToolboxNameErr	Une ressource du système manque
-915	noResponseErr	Impossible de contacter l'application de destination
-916	portClosedErr	Port fermé
-917	sessClosedErr	Session fermée
-919	badPortNameErr	PPCPortRec est invalide
-922	noDefaultUserErr	L'utilisateur n'a pas spécifié le nom du propriétaire dans le panneau de contrôle de configuration de partage
-923	notLoggedInErr	La valeur par défaut de userRefNum n'existe pas encore
-924	noUserRefErr	Impossible de créer un nouveau userRefNum
-925	networkErr	Une erreur s'est produite dans le réseau
-926	noInformErr	PPCStart a échoué parce que la destination n'a pas émis d'informations
-927	authFailErr	Le mot de passe de l'utilisateur est incorrect

Tableau A-7. Codes d'erreur PPC (Suite)

Code	Nom	Description
-928	noUserRecErr	Le numéro de référence de l'utilisateur est invalide
-930	badServiceMethodErr	La méthode de service est différente de PPCServiceRealTime
-931	badLocNameErr	Le nom de l'emplacement est invalide
-932	guestNotAllowedErr	Le port de destination requiert une authentification

Tableau A-8. Codes d'erreur GPIB

Code d'erreur	Nom de l'erreur	Description
0	EDVR	Erreur lors de la connexion au driver
1	ECIC	La commande requiert que le contrôleur GPIB soit CIC (contrôleur en charge)
2	ENOL	L'écriture n'a pas détecté d'auditeurs
3	EADR	Le contrôleur GPIB n'est pas correctement adressé
4	EARG	Argument(s) invalide(s)
5	ESAC	La commande requiert que le contrôleur GPIB soit le SC
6	EABO	Opération d'E/S abandonnée
7	ENEB	Carte inexistante
8	EDMA	Erreur matérielle DMA détectée
9	EBTO	Timeout de bus μ P de matériel DMA
11	ECAP	Aucune capacité
12	EFSO	Erreur dans l'opération du système de fichier
13	EOWN	Carte partageable possédée en exclusivité
14	EBUS	Erreur du bus GPIB
15	ESTB	Débordement de la file d'attente d'octets d'interrogation série
16	ESRQ	SRQ bloqué
17	ECMD	Commande non reconnue
19	EBNP	Carte absente
20	ETAB	Erreur de tableau

Tableau A-8. Codes d'erreur GPIB (Suite)

Code d'erreur	Nom de l'erreur	Description
30	NADDR	Aucune entrée d'adresse GPIB
31	NSTRG	Aucune entrée de chaîne de caractères (écriture)
32	NCNT	Aucune entrée de comptage (lecture)

Tableau A-9. Codes d'erreur des fonctions LabVIEW

Code d'erreur	Nom de l'erreur	Description
0	—	Aucune erreur
1	—	Erreur d'argument du gestionnaire
2	—	Erreur d'argument
3	—	Hors zone
4	—	Fin de fichier
5	—	Fichier déjà ouvert
6	—	Erreur d'E/S sur fichier générique
7	—	Fichier non trouvé
8	—	Erreur d'autorisation de fichier
9	—	Disque plein
10	—	Chemin dupliqué
11	—	Trop de fichiers ouverts
12	—	Fonction du système non activée
13	—	Fichier de ressources non trouvé
14	—	Impossible d'ajouter des ressources
15	—	Ressource non trouvée
16	—	Image non trouvée
17	—	Erreur de mémoire image
18	—	Le stylo n'existe pas
19	—	Type de configuration invalide
20	—	Symbole de configuration non trouvé
21	—	Erreur de configuration de l'analyse
22	—	Erreur de configuration de la mémoire

Tableau A-9. Codes d'erreur des fonctions LabVIEW (Suite)

Code d'erreur	Nom de l'erreur	Description
23	—	Format du code externe incorrect
24	—	Offset du code externe incorrect
25	—	Code externe absent
26	—	Fenêtre nulle
27	—	Erreur de destruction de fenêtre
28	—	Menu nul
29	—	Impression abandonnée
30	—	Enregistrement d'impression incorrect
31	—	Erreur d'impression du driver
32	—	Erreur de fenêtres pendant l'impression
33	—	Erreur de mémoire pendant l'impression
34	—	Erreur du dialogue d'impression
35	—	Erreur d'impression générique
36	—	Refnum du périphérique invalide
37	—	Périphérique non trouvé
38	—	Erreur du paramètre du périphérique
39	—	Erreur de l'unité de périphérique
40	—	Impossible d'ouvrir le périphérique
41	—	Appel du périphérique abandonné
42	—	Erreur générique
43	—	Annulé par l'utilisateur
44	—	ID d'objet trop bas
45	—	ID d'objet trop haut
46	—	Objet non inclus dans l'ensemble
47	—	Ensemble inconnu
48	—	Objet inconnu (DefProc invalide)
49	—	Objet inconnu (DefProc pas dans le tableau)
50	—	Message hors gamme
51	—	Méthode invalide (nulle)

Tableau A-9. Codes d'erreur des fonctions LabVIEW (Suite)

Code d'erreur	Nom de l'erreur	Description
52	—	Message inconnu
53	—	Appel de gestionnaire non supporté
54	—	Adresse incorrecte
55	—	Connexion en cours
56	—	Connexion différée
57	—	Connexion déjà en cours
58	—	Attribut réseau non supporté
59	—	Erreur réseau
60	—	Adresse en cours d'utilisation
61	—	Système à court de mémoire
62	—	Connexion abandonnée
63	—	Connexion refusée
64	—	Non connecté
65	—	Déjà connecté
66	—	Connexion fermée
67	—	Erreur d'initialisation (gestionnaire interapplications)
68	—	Occurrence incorrecte
69	—	Gestionnaire d'attente d'une occurrence non limitée
70	—	Débordement de la file d'attente d'occurrence
71	—	Conflit des types d'enregistrement de données
72	—	Inutilisé
73	—	Type non reconnu (gestionnaire d'interapplications)
74	—	Mémoire altérée
75	—	DLL temporaire non effectuée
76	—	Version de CIN ancienne
77	—	Code d'erreur inconnu
81	—	Mauvaise correspondance du type de spécification de format
82	—	Spécification de format inconnue

Tableau A-9. Codes d'erreur des fonctions LabVIEW (Suite)

Code d'erreur	Nom de l'erreur	Description
83	—	Pas assez de spécifications de format
84	—	Trop de spécifications de format
85	—	Echec du balayage

Tableau A-10. Codes d'erreur PPC spécifiques à LabVIEW

Code d'erreur	Nom d'erreur	Description
1	errNoPPCToolBox	La PPC Toolbox n'existe pas (requiert System 7.0 ou une version plus récente), ou n'a pas pu être initialisée
2	errNoGlobals	Le CIN du VI PPC n'a pas pu obtenir ses variables globales
3	errTimedOut	L'opération PPC a dépassé sa limite de timeout
4	errAuthRequired	La cible spécifiée dans le VI PPC Start Session requiert une authentification, mais le dialogue d'authentification n'a pas été autorisé
5	errbadState	Etat du VI PPC Start Session inattendu

Tableau A-11. Codes d'erreur TCP et UDP

Code d'erreur	Nom d'erreur	Description
53	mgNotSupported	L'appel du gestionnaire de LabVIEW n'est pas supporté
54	ncBadAddressErr	Adresse réseau mal libellée
55	ncInProgressErr	Opération en cours
56	ncTimeOutErr	L'opération a dépassé la limite temporelle spécifiée par l'utilisateur
57	ncBusyErr	Connexion occupée
58	ncNotSupportedErr	Fonction non supportée
59	ncNetErr	Le réseau est en panne, ne peut pas être atteint ou n'a pas été mis à zéro
60	ncAddrInUseErr	L'adresse spécifiée est actuellement utilisée
61	ncSysOutOfMemErr	Le système n'a pas pu allouer la mémoire nécessaire
62	ncSysConnAbortedErr	Le système a entraîné l'abandon de la connexion
63	ncConnRefusedErr	La connexion n'est pas établie

Tableau A-11. Codes d'erreur TCP et UDP (Suite)

Code d'erreur	Nom d'erreur	Description
65	ncAlreadyConnectedErr	La connexion est déjà établie
66	ncConnClosedErr	La connexion a été fermée par un pair

Tableau A-12. Codes d'erreur des ports série

Code d'erreur	Nom d'erreur	Description
61	EPAR	Erreur de parité du port série
62	EORN	Erreur de surcharge du port séri
63	EOFLL	Le port série reçoit un débordement de buffer.
64	EFRM	Erreur de cadrage du port série
65	SPTMO	Timeout du port série, octets non reçus au port série

Tableau A-13. Codes d'erreur des messages AppleEvent spécifiques à LabVIEW

Code d'erreur	Nom d'erreur	Description
1000	klVE_InvalidState	L'état du VI ne lui permet pas de fonctionner
1001	klVE_FPNOpen	La face-avant du VI n'est pas ouverte
1002	klVE_CtrlErr	Des commandes de la face-avant du VI sont à l'état d'erreur.
1003	klVE_VIBad	Le VI est brisé
1004	klVE_NotInMem	Le VI n'est pas en mémoire

Tableau A-14. Codes d'erreur DDE

Code d'erreur	Nom de l'erreur	Description
00000	—	Aucune erreur
14001	DDE_INVALID_REFNUM	Refnum invalide
14002	DDE_INVALID_STRING	Chaîne de caractères invalide
14003	DDEML_ADVACKTIMEOUT	La demande de transfert d'informations synchrone est en timeout
14004	DDEML_BUSY	La réponse définit le bit DDE_FBUSY
14005	DDEML_DATAACKTIMEOUT	La demande d'un transfert de données synchrone est en timeout

Tableau A-14. Codes d'erreur DDE (Suite)

Code d'erreur	Nom de l'erreur	Description
14006	DDEML_DDL_NOT_INITIALIZED	DDEML a effectué un appel sans commencer par appeler <code>DdeInitialize</code> , ou a reçu un identificateur d'instance invalide
14007	DDEML_DLL_USAGE	Une application de contrôle ou de client seul a tenté d'effectuer un transfert DDE
14008	DDEML_EXECACKTIMEOUT	La demande de transfert d'exécution synchrone est en timeout
14009	DDEML_INVALIDPARAMETER	Paramètre non validé pas le DDEML
14010	DDEML_LOW_MEMORY	L'application du serveur a surchargé le client, en occupant beaucoup de place dans la mémoire
14011	DDEML_MEMORY_ERROR	Une affectation de la mémoire a échoué
14012	DDEML_NOTPROCESSED	La demande ou le "poke" est destiné(e) à un article invalide
14013	DDEML_NO_CONV_ESTABLISHED	La tentative de dialogue du client a échoué
14014	DDEML_POKEACTIMEOUT	Le transfert a échoué
14015	DDEML_POSTMSG_FAILED	La demande de transfert de "poke" synchrone s'est arrêtée
14016	DDEML_REENTRANCY	Une application avec un transfert synchrone en cours a tenté de démarrer un autre transfert, ou une fonction de callback DDEML a appelé <code>DdeEnableCallback</code>
14017	DDEML_SERVER_DIED	Le transfert côté serveur tenté lors du dialogue a été arrêté par le client, ou le service s'est arrêté avant de finir une transaction
14018	DDEML_SYS_ERROR	Erreur interne dans le DDEML
14019	DDEML_UNADVACKTIMEOUT	La demande d'arrêt des informations est en timeout
14020	DDEML_UNFOUND_QUEUE_ID	Identificateur de transfert invalide envoyé à la fonction DDEML
14021	—	Code de commande invalide
14022	—	Timeout d'occurrence ; le transfert s'est arrêté avant d'être terminé

Capacités matérielles DAQ

Cette annexe contient des tableaux qui résument les capacités d'E/S analogiques et numériques des périphériques d'acquisition de données (DAQ) de National Instruments. Les périphériques présentés dans cette annexe sont groupés en catégories. Les catégories de périphériques DAQ contenus dans ces tableaux sont les suivantes :

- Les périphériques MIO et AI
- Les périphériques Série 1200, Lab et portables
- Les périphériques Série 54xx
- Les modules SCXI
- Les périphériques d'acquisition de signaux dynamiques (Dynamic Signal Acquisition Devices)
- Les périphériques de sorties analogiques uniquement (Analog Output Only Devices)
- Les périphériques numériques uniquement (Digital Only Devices)
- Les périphériques de séquençement uniquement (Timing Only Devices)
- Les capacités matérielles des périphériques 5102



Remarque (Macintosh) *Lorsqu'un périphérique NuBus indique qu'il supporte des transferts DMA, un périphérique DMA (tel qu'un NB-DMA2800) est également requis.*

Capacités matérielles des périphériques MIO et AI

Tableau B-1. Programmabilité de la configuration d'entrées analogiques—Périphériques MIO et AI

Périphérique	Gain	Gamme	Polarité	Asym/DIFF	Couplage
Tous les périphériques MIO Série E Tous les périphériques AI Série E	Par voie	Par voie	Par voie	Par voie	DC AC, DC (pour PCI-6110E, PCI-6111E)
AT-MIO-16F-5	Par voie	Par groupe	Par groupe	Par groupe	DC
AT-MIO-64F-5 AT-MIO-16X	Par voie	Par voie	Par voie	Par voie	DC
AT-MIO-16/16D NB-MIO-16 NB-MIO-16X	Par voie	Par périphérique	Par périphérique	Par périphérique	DC

Par périphérique signifie que vous pouvez sélectionner la valeur d'un paramètre avec des cavaliers matériels et que la sélection affecte tous les groupes de voies du périphérique. *Par groupe* signifie que vous pouvez programmer la sélection via le logiciel et que celle-ci affecte toutes les voies utilisées au même moment. *Par voie* signifie que vous pouvez programmer la sélection avec des cavaliers matériels ou via le logiciel voie par voie. Lorsqu'une valeur spécifique d'un paramètre est affichée, elle est enregistrée.

Tableau B-2. Caractéristiques des entrées analogiques—Périphériques MIO et AI (1^{ère} partie)

Périphérique	Nombre de voies	Résolution	Gains ¹	Gamme (V) ¹	Entrée FIFO (mots)	Balayage ²
AT-MIO-16E-1 AT-MIO-16E-2 AT-MIO-16E-10 AT-MIO-16DE-10 NEC-MIO-16E-4 PCI-MIO-16E-1 PCI-MIO-16E-4 NEC-AI-16E-4	16 Asym, 8 DI	12 bits	0,5, 1, 2, 5, 10, 20, 50, 100	±5, 0 à 10	512; E-1: 8,192; E-2 et E-4: 2,048	Jusqu'à 512
PCI-6110E PCI-6111E	4 DI 2 DI	12 bits 12 bits	0,2, 0,5, 1,2, 5, 10, 20, 50	±10	512	Jusqu'à 4 Jusqu'à 2

Tableau B-2. Caractéristiques des entrées analogiques—Périphériques MIO et AI (1^{ère} partie) (Suite)

Périphérique	Nombre de voies	Résolution	Gains ¹	Gamme (V) ¹	Entrée FIFO (mots)	Balayage ²
AT-MIO-64E-3*	64 Asym, 32 DI	12 bits	0,5, 1, 2, 5, 10, 20, 50, 100	±5, 0 à 10	2048	Jusqu'à 512
PCI-MIO-16XE-10	16 Asym, 8 DI	16 bits	1, 2, 5, 10, 20, 50,100	±10, 0 à 10	512	Jusqu'à 512
NEC-MIO-16XE-50 NEC-AI-16XE-50 AT-MIO-16XE-50 DAQPad-MIO-16XE-50 PCI-MIO-16XE-50	16 Asym, 8 DI	16 bits	1, 2, 10, 100	±10, 0 à 10	512	Jusqu'à 512
AT-MIO-16F-5 AT-MIO-64F-5**	16 Asym, 8 DI 64 Asym, 32 DI	12 bits	0,5, 1, 2, 5, 10, 20, 50, 100	±5, ±10, 0 à 10	16F-5 : 256; 64F-5 : 512	Jusqu'à 512
AT-MIO-16X	16 Asym, 8 DI	16 bits	1, 2, 5, 10, 20, 50, 100	±10, 0 à 10	512	Jusqu'à 512
AT-MIO-16(L) AT-MIO-16(H) AT-MIO-16D(L) AT-MIO-16D(H)	16 Asym, 8 DI	12 bits	(L) 1, 10, 100, 500 ; (H) : 1, 2, 4, 8	±5, ±10, 0 à 10	16 (L,H); 512 (DL, DH)	Jusqu'à 16
NB-MIO-16 NB-MIO-16X	16 Asym, 8 DI	MIO-16 : 12 ; MIO-16X : 16	(L) 1, 10, 100, 500 ; (H) 1, 2, 4, 8	±10, ±5, 0 à 10, 0 à 5	16 ; MIO-16, Rév. G : 512	Jusqu'à 16; MIO-16 : groupes de 2, 4, 8, et 16

¹ Vous pouvez déterminer les paramètres limites de votre périphérique en multipliant les valeurs de gamme et de tension. Pour plus d'informations sur les paramètres limites dans LabVIEW, reportez-vous au chapitre 3, *Concepts de base de l'acquisition de données LabVIEW* dans le *Manuel de base d'acquisition de données LabVIEW*.

² Balayage = voie, dans n'importe quel ordre.

* Les voies valides pour le périphérique AT-MIO-64E-3 en mode différentiel sont 0–7, 16–23, 32–39 et 48–55.

** Les voies valides pour le périphérique AT-MIO-64F-5 en mode différentiel sont 0–7 et 16–39.

Tableau B-3. Caractéristiques des entrées analogiques—Périphériques MIO et AI (2^{ème} partie)

Périphérique	Déclenchements ¹	Fréquence d'échantillonnage max (E/s)	Méthode de transfert
AT-MIO-16E-1 AT-MIO-16E-2 AT-MIO-64E-3 AT-MIO-16E-10 AT-MIO-16DE-10 PCI-MIO-16E-1 PCI-MIO-16XE-10 NEC-AI-16E-4 NEC-MIO-16E-4 PCI-MIO-16E-4 PCI-6110E PCI-6111E	Logiciel, Pré, Post, (et analogique sur E-1, E-2, E-3, E-4, PCI-6110E, et PCI-6111E)	E-1: 1 M, E-2 et E-3 : 500 k, E-4 : 250 k, E-10 et DE-10 : 100 k PCI-6110E et PCI-6111E : 5M	DMA, interruptions
Tous les périphériques MIO-16XE-50 NEC-AI-16XE-50	Logiciel, Pré, Post	20 k	DMA, (interruptions sur DAQPad-MIO-16XE-50)
AT-MIO-16F-5 AT-MIO-64F-5	Logiciel, Pré, Post	200 k	DMA, interruptions
AT-MIO-16X AT-MIO-16/16D	Logiciel, Pré, Post	100 k	DMA, interruptions
NB-MIO-16	Logiciel, Post	111 k (L-9 ou H-9), 67 k (L-15 ou H-15), 40 k (L-25 ou H-25)	DMA, interruptions
NB-MIO-16X	Logiciel, Post	55 k (L-18 ou H-18), 24 k (L-42 ou H-42)	DMA, interruptions

¹ Logiciel = Déclenchement logiciel (également appelé récupération conditionnelle), Pré = Pré-déclenchement, Post = Post-déclenchement.



Remarque *Pour les périphériques NB-MIO, le déclenchement logiciel est en fait effectué dans la routine d'interruption de service (interruptions uniquement) et est différent de la récupération conditionnelle.*

Tableau B-4. Support de voies internes—Périphériques MIO et AI

Périphérique	Voies internes
AT-MIO-16XE10, AT-MIO-16XE-50, NEC-MIO-16XE-50, DAQPad-MIO-16XE-50	IntAIGnd, IntRef5V, IntAOGndVsAIGnd, IntAOCh0, IntAOCh0VsRef5V, IntA0Ch1, IntAOCh1VsRef5V
DAQCard-AI-16E-4, NEC-AI-16E-4	IntAIGnd, IntRef5V, IntA0GndVsAIGnd
PCI-MIO-16XE-10, PCI-MIO-16XE-50, PXI-6030E, PXI-6011E, PCI-6031, CPCI-6030E, CPCI-6011E, VXI-MIO-64XE-10	IntAIGnd, IntRef5V, IntAOGndVsAIGnd, IntAOch0, IntAOCh0VsRef5V, IntAOCh1, IntAOCh1VsRef5V, IntAOChVsAOch0, IntDevTemp
PCI-MIO-16E-1, PCI-MIO-16E-4, PXI-6070E, PXI-6040E, PCI-6071E, CPCI-6070E, CPCI-6040E, VXI-MIO-64E-1	IntAIGnd, IntRef5V, IntCmRef5V, IntAOGndVsAIGnd, IntAOCh0, IntAOCh0VsRef5V, IntAOCh1, IntAOCh1VsRef5V, IntAOCh1VsAOCh0, IntDevTemp
AT-AI-16XE-10, PCI-6032E, PCI-6033E, DAQCard-AI-16XE-50, NEC-AI-16XE-50	IntAIGnd, IntRef5V, IntA0GndVsAIGnd
AT-MIO-16E-1, AT-MIO-16E-2, AT-MIO-16E-3, AT-MIO-16DE-10, AT-MIO-16E-10, DAQPad-6020E, NEC-MIO-16E-4	IntAIGnd, IntRef5V, IntCmRef5V, IntAOGndVsAIGnd, IntAOCh0, IntAOCh0VsRef5V, IntAOCh1, IntAOCh1VsRef5V

Tableau B-5. Caractéristiques des sorties analogiques—Périphériques MIO et AI

Périphérique	Numéro des voies	Type de C/N/A	Taille du FIFO	Limites de sortie (V)	Horloges de mise à jour	Méthode de transfert
AT-MIO-16E-1 AT-MIO-16E-2 AT-MIO-64E-3 NEC-MIO-15E-4 VXI-MIO-64E-1	0, 1	12 bits double buffer	2048	0 à 10, ± 10 , $\pm V_{\text{réf}}$, 0 à $V_{\text{réf}}$	Horloge de mise à jour 1 ou mise à jour externe	DMA, interruptions
AT-MIO-16E-10	—	12 bits double buffer	0	± 10 , 0 à 10	Horloge de mise à jour 1 ou mise à jour externe	DMA, interruptions
AT-MIO-XE-50 NEC-MIO-16XE-50	—	12 bits double buffer	0	± 10	Horloge de mise à jour 1 ou mise à jour externe	DMA, interruptions

Tableau B-5. Caractéristiques des sorties analogiques—Périphériques MIO et AI (Suite)

Périphérique	Numéro des voies	Type de C/N/A	Taille du FIFO	Limites de sortie (V)	Horloges de mise à jour	Méthode de transfert
DAQPad-MIO-16XE-50	—	12 bits double buffer	0	± 10	Horloge de mise à jour 1 ou mise à jour externe	Interruptions
AT-MIO-16XE-10 VXI-MIO-64XE-10	—	16 bits double buffer	0	± 10 , 0 à 10	Horloge de mise à jour 1 ou mise à jour externe	DMA, interruptions
OCI-MIO-16E-1 CPCI-6070E PXI-6070E PCI-6071E	—	12 bits double buffer	2048	± 10 , 0 à 10	Horloge de mise à jour 1 ou mise à jour externe	DMA, interruptions
PCI-MIO-16E-4 CPCI-6040E PXI-6040E	—	12 bits double buffer	512	± 10 , 0 à 10	Horloge de mise à jour 1 ou mise à jour externe	DMA, interruptions
PCI-MIO-16XE-50 CPCI-6011E PXI-6011E	—	12 bits double buffer	0	± 10	Horloge de mise à jour 1 ou mise à jour externe	DMA, interruptions
PCI-MIO-16XE-10 CPCI-6030E PXI-6030E PCI-6031E	—	16 bits double buffer	5048	± 10 , 0 à 10	Horloge de mise à jour 1 ou mise à jour externe	DMA, interruptions
DAQPad-6020E	—	12 bits double buffer	0	± 10 , 0 à 10	Horloge de mise à jour 1 ou mise à jour externe	DMA, interruptions
PCI-6110E PCI-6111E	—	16 bits double buffer	4096	± 5	Horloge de mise à jour 1 ou mise à jour externe	DMA, interruptions

Tableau B-5. Caractéristiques des sorties analogiques—Périphériques MIO et AI (Suite)

Périphérique	Numéro des voies	Type de C N/A	Taille du FIFO	Limites de sortie (V)	Horloges de mise à jour	Méthode de transfert
AT-MIO-16F-5 AT-MIO-64F-5	0, 1	12 bits double buffer (64F-5 : FIFO de 2K)	—	0 à 10, ± 10 , $\pm V_{\text{réf}}$, 0 à $V_{\text{réf}}$	Première horloge de mise à jour 1 d'abord disponible sur ctr 5, 2, 1 ou mise à jour externe. La valeur par défaut est 5. La gamme des signaux de base de temps est 5 000 000, 1 000 000, 100 000, 10 000, 1000 et 100	DMA, interruptions
AI-MIO-16X	0, 1	16 bits double buffer (FIFO de 2K)	—	± 10 , 0 à 10, $\pm V_{\text{réf}}$, 0 à $V_{\text{réf}}$	Première horloge de mise à jour 1 disponible sur ctr 5, 2, 1 ou mise à jour externe. La valeur par défaut est 5. La gamme des signaux de base de temps est 5 000 000, 1 000 000, 100 000, 10 000, 1000 et 100	DMA, interruptions

Tableau B-5. Caractéristiques des sorties analogiques—Périphériques MIO et AI (Suite)

Périphérique	Numéro des voies	Type de C N/A	Taille du FIFO	Limites de sortie (V)	Horloges de mise à jour	Méthode de transfert
AT-MIO-16/16D	0, 1	12 bits double buffer	—	0 à 10, ± 10 , $\pm V_{\text{réf}}$, 0 à $V_{\text{réf}}$	L'horloge de mise à jour 1 est ctr 2 ou mise à jour externe. La gamme des signaux de base de temps est 1 000 000, 100 000, 10 000, 1 000 et 100.	Interruptions
NB-MIO-16/16X	0, 1	MIO-16 : 12 bits ; MIO-16X : 12 bits double buffer	—	0 à 10, ± 10 , $\pm V_{\text{réf}}$, 0 à $V_{\text{réf}}$	Horloge de mise à jour 1, mise à jour externe (MIO-16X uniquement). La gamme des signaux de base de temps est 1 000 000, 100 000, 10 000, 1 000 et 100.	MIO-16:DMA ; MIO-16X : DMA, interruptions

Tableau B-6. Caractéristiques des sorties analogiques—Périphériques Série E

Périphérique	Capacité à redéfinir les signaux	Capacité à référencer la mise à la masse	Peut contrôler les modes de demande FIFO	Fenêtrage des AO, supportant les pauses/reprises	Plusieurs buffers supportés
AT-MIO-16E-1 AT-MIO-16E-2 AT-MIO-64E-3 NEC-MIO-16E VXI-MIO-64E-1	Oui	Non	Non	Non	Oui
AT-MIO-16E-10 AT-MIO-16DE-10	Non	Non	Non	Oui	Oui
AT-MIO-XE-50 NEC-MIO-16XE-50	Non	Non	Non	Non	Oui
DAQPad-MIO-16XE-50	Non	Non	Non	Non	Oui
AT-MIO-16XE-10	Non	Non	Non	Oui	Oui
VXI-MIO-64XE-10	Non	Non	Non	Non	Oui
PCI-MIO-16E-1 CPCI-6070E PXI-6070E PCI-6071E	Oui	Oui	Oui	Oui	Non
PCI-MIO-16E-4 CPCI-6040E PXI-6011E	Non	Oui	Oui	Oui	Non
PCI-MIO-16XE-10 CPCI-6030E PXI-6030E PCI-6031E	Non	Non	Oui	Oui	Non
DAQPad-6020E	Non	Oui	Non	Oui	Non
PCI-6110E PCI-6111E	Non	Non	Oui	Oui	Non

Tableau B-7. Capacités matérielles des E/S numériques—Périphériques MIO et AI

Périphérique	Type de port	Numéros de port	Modes de handshaking	Direction	Horloges DIO	Méthode de transfert
Tous les périphériques MIO-16E AT-MIO-16D ¹ AT-MIO-64F-5	Ports 4 bits	0, 1	Aucun handshaking	Lecture ou écriture	Aucune	Interrogation logicielle
Tous les périphériques MIO-16E Tous les périphériques de série NEC-E AT-MIO-64E-12 AT-MIO-16DE-10 ¹ AT-MIO-16XE-50 DAQPad-MIO-16XE-50 PCI-MIO-16XE-50 PXI-6040E (MIO-16E-4) PXI-6070E (MIO-16E-1) PXI-6071E (MIO-64E-1) PCI-6031E (MIO-64XE-10) PCI-6032E (AI-16XE-10) PCI-6033E (AI-64XE-10) PCI-6110E/PCI-6111E	Ports 8 bits	0	Aucun handshaking	Contrôle de la direction en représentation binaire	Aucune	Interrogation logicielle
AT-MIO-16D ¹ AT-MIO-16DE-10 ¹	Ports 8 bits	2, 3	Handshaking activé ou désactivé	Lecture ou écriture, le port 2 peut être bi-directionnel	Aucune	Interruptions
	Ports 8 bits	4	Aucun handshaking ; inutilisable si le port 2 ou 3 utilise le handshaking	Lecture ou écriture	Aucune	Interrogation logicielle

¹ Ces périphériques apparaissent plusieurs fois dans ce tableau parce qu'ils ont des fonctionnalités numériques avancées.

Tableau B-8. Caractéristiques des compteurs—Périphériques MIO et AI

Périphérique	Circuit de compteur utilisé	Nombre de compteurs généraux disponibles	Bases de temps disponibles	Nombre de bits	Modes de fenêtrage disponible	Sorties disponibles	Modes de sortie disponibles	Direction de comptage ¹
Périphériques Série E	DAQ-STC	2	2 internes : 20 mégahertz ou 100 kilohertz ; externe	24	front montant, front descendant, niveau haut, niveau bas	2		montante ou descendante, peut-être contrôlée par logiciel ou matériel
AT-MIO-16F-5 AT-MIO-64F-5 AT-MIO-16/16D NB-MIO-16/16X	Am-9513	3	5 ou 6 internes : 5 mégahertz (uniquement sur CTR2 de 16F-5, 64F-5 et AT-MIO-16X), 1 mégahertz, 100 kilohertz, 10 kilohertz, 1 kilohertz, 100 Hz ; externe	16	front montant, front descendant, niveau haut, niveau bas	2	impulsion TC ou bascule TC	montante

Tableau B-9. Utilisation des compteurs pour entrées et sorties analogiques—Périphériques MIO et AI

Nom de périphérique	Circuit de compteur utilisé	Horloge d'entrée analogique de voie	Compteur d'échantillonnage	Horloge de balayage analogique de voie	Horloge de mise à jour de sortie analogique
Périphériques Série E	DAQ-STC	Le circuit DAQ-STC utilise des horloges spécifiques destinées à ces fonctions.			
AT-MIO-16F-5 AT-MIO-64F-5 AT-MIO-16X	Am9513	Ctr 3	Ctr 4 (& 5) ¹	Ctr 2 (ou 1) ²	Ctr 5, 2 ou 1
AT-MIO-16/16D NB-MIO-16X	Am9513	Ctr 3	Ctr 4 (& 5) ¹	Ctr 2 (ou 1) ²	Ctr 2 (et via DMA pour NB-MIO-16X)
NB-MIO-16	Am9513	Ctr 3	Ctr 4 (& 5) ¹	Aucune (ou 1) ²	(via DMA)

¹ Si le nombre total d'échantillons est inférieur à 65 535, seul le premier compteur est utilisé. Si ce nombre dépasse 65 536, le premier compteur est utilisé avec le deuxième compteur comme un compteur d'échantillons 32 bits.

² Ctr 2 (ou aucun compteur pour NB-MIO-16) est utilisé pour des opérations de balayage normales et Ctr 1 est utilisé pour le balayage matériel de l'AMUX-64T et de SCXI.

Capacités matérielles des périphériques Série 1200/Lab et portables

Tableau B-10. Programmabilité de la configuration des entrées analogiques—Périphériques Série 1200/Lab et portables

Périphérique	Gain	Gamme	Polarité	Asym/DIFF	Couplage
Lab-LC Lab-NB	Par groupe	Par périphérique	Par périphérique	Asym	DC
Lab-PC+	Par groupe	Par groupe	Par périphérique	Par périphérique	DC
SCXI-1200 DAQPad-1200 DAQCard-1200 PCI-1200	Par groupe	Par groupe	Par groupe	Par groupe	DC
DAQCard-500	1	1 gamme disponible uniquement	Bipolaire	Asym	DC
DAQCard-PC-516	1	1 gamme disponible uniquement	Bipolaire	Par groupe	DC
DAQCard-700	1	Par groupe	Bipolaire	Par groupe	DC
PC-LPM-16	1	Par périphérique	Bipolaire	Asym	DC


 **Remarque** *Par périphérique signifie que vous pouvez sélectionner la valeur d'un paramètre avec des cavaliers matériels et que la sélection affecte tous les groupes de voies du périphérique. Par groupe signifie que vous pouvez programmer la sélection via le logiciel et que celle-ci affecte toutes les voies utilisées au même moment. Par voie signifie que vous pouvez programmer la sélection avec des cavaliers matériels ou via le logiciel voie par voie. Lorsque la valeur spécifique d'un paramètre est affichée, elle est enregistrée.*

Tableau B-11. Caractéristiques des entrées analogiques—Périphériques Série 1200/Lab et portables (1^{ère} partie)

Périphérique	Nombre de voies	Résolution (bits)	Gains ¹	Gamme (V) ¹	Entrée FIFO (échantillons)
Lab-LC Lab-NB	8 Asym	12	1, 2, 5, 10, 20, 50, 100	±5, 0 à 10	16
Lab-PC+ SCXI-1200 DAQPad-1200 DAQCard-1200 PCI-1200	8 Asym, 4 DI	12	1, 2, 5, 10 20, 50, 100	±5, 0 à 10	2 048 ; Lab-PC : 512
DAQCard-500	8 Asym	12	1	±5	16
DAQCard PC516	8 Asym, 4 DI	16	1	±5	512
DAQCard-700	16 Asym, 8 DI	12	1	±10, ±5, ±2,5	512
PC-LPM-16	16 Asym	12	1	±5, ±2,5, 0 à 10, 0 à 5	16

¹ Vous pouvez déterminer les paramètres limites de votre périphérique en multipliant les valeurs de gamme et de tension. Pour plus d'informations sur les paramètres limites dans LabVIEW, reportez-vous au chapitre 3, *Concepts de base de l'acquisition de données LabVIEW* dans le *Manuel de base d'acquisition de données LabVIEW*.

Tableau B-12. Caractéristiques des entrées analogiques—Périphériques Série 1200/Lab et portables (2^{ème} partie)

Périphérique	Balayage	Déclenchements	Fréquence d'échantillonnage max (E/s)	Méthode de transfert
Lab-LC Lab-NB	N'importe quelle voie unique ; pour plusieurs voies, de N à 0, où $N \leq 7$	Déclenchement logiciel, pré-déclenchement et post-déclenchement avec déclenchement numérique	62,5 k	Interruptions
Lab-PC+ SCXI-1200 DAQPad-1200 DAQCard-1200 PCI-1200	N'importe quelle voie unique ; pour plusieurs voies, de N à 0, où $N \leq 7$	Déclenchement logiciel, pré-déclenchement et post-déclenchement avec déclenchement numérique	100 k ; Lab-PC+ : 83 k	Interruptions ; Lab-PC+ : Interruptions, DMA
DAQCard-500 DAQCard 516 PC-516	N'importe quelle voie unique ; pour plusieurs voies, de N à 0, où $N \leq 7$	Déclenchement logiciel uniquement	50 k	Interruptions
DAQCard-700	N'importe quelle voie unique ; pour plusieurs voies, de N à 0, où $N \leq 15$	Déclenchement logiciel uniquement	100 k	interruptions
PC-LPM-16	N'importe quelle voie unique ; pour plusieurs voies, de N à 0, où $N \leq 15$	Déclenchement logiciel uniquement	50 k	Interruptions

Tableau B-13. Caractères des sorties analogiques—Périphériques Série 1200/Lab et portables

Périphérique	Nombre de voies	Type de C N/A	Limites des sorties (V)	Horloges de mise à jour	Groupe ment des formes d'ondes	Méthodes de transfert
Lab-NB Lab-LC	0, 1	12 bits double buffer	0 à 10, ± 5	L'horloge de mise à jour 1 est ctr A2 ou mise à jour externe ; la base de temps est 1 mégahertz ou ctrB0	0, 1 ou 0 et 1	Interruptions
Lab-PC+ SCXI-1200 DAQPad-1200 DAQCard-1200 PCI-1200	0, 1	12 bits double buffer	0 à 10, ± 5	L'horloge de mise à jour 1 est ctr A2 ou mise à jour externe ; la gamme de signal de base de temps est 1 000 000, 100 000, 10 000, 1 000 et 100	0, 1 ou 0 et 1	Interruptions


 **Remarque** Les périphériques DAQCard-516 et PC 516 n'ont pas de sorties analogiques.

Tableau B-14. Usage des compteurs pour les entrées et les sorties analogiques—
Périphériques Série 1200/Lab et portables

Nom du périphérique	Circuit de compteur utilisé	Horloge de voie d'entrée analogique	Compteur d'échantillonnage d'entrée analogique	Horloge de balayage d'entrée analogique	AO Horloge de mise à jour de sortie analogique
Lab-NB, Lab-LC	82C53	Ctr A0 (& B0) ¹	Ctr A1	Aucune	Ctr A2
Lab-PC+, DAQPad-1200, SCXI-1200, DAQCard-1200, PCI-1200	82C53	Ctr A0 (& B0) ¹	Ctr A1	Ctr B1	Ctr A2
DAQCard-500, DAQCard-700,	8254	Ctr 0	(logiciel)	Aucune	Aucune
DAQCard 516 PC-516	82C54	Ctr 0	Logiciel	Aucune	Aucune
PC-LPM-16	82C53	Ctr 0	(logiciel)	Aucune	Aucune

¹ Le deuxième compteur est utilisé comme une extension de base de temps pour les entrées et les sorties analogiques séquencées lorsque l'intervalle d'échantillonnage dépasse 65 535 ms.

Tableau B-15. Capacités matérielles des E/S numériques—Périphériques Série 1200/Lab et portables

Périphérique	Type de port	Numéros de port	Modes de handshaking	Direction	Horloges DIO	Méthode de transfert
Lab-NB Lab-LC Lab-PC+ SCXI-1200 DAQCard-1200 DAQPad-1200 PCI-1200	Port 8 bits	0, 1	Handshaking activé ou désactivé	Lecture ou écriture, le port 0 peut être bidirectionnel	Aucune	Interruptions
	Port 8 bits	2	Aucun handshaking ; inutilisable si le port 0 ou 1 utilise le handshaking	Lecture ou écriture	Aucune	Interrogation logicielle
PC-LPM-16	Ports 8 bits	0, 1	Aucun handshaking	0 : lecture ou écriture	Aucune	Interrogation logicielle
DAQCard-500 DAQCard-516	Ports 4 bits	0, 1	Aucun handshaking	0 : écriture, 1 : lecture	Aucune	Interrogation logicielle
DAQCard-700	Ports 8 bits	0, 1	Aucun handshaking	0 : écriture 1 : lecture	Aucune	Interrogation logicielle

Périphériques 54xx

Tableau B-16. Caractéristiques des sorties analogiques et numériques—Périphériques Série 54XX

Caractéristiques	AT-5411, PCI-5411
Nombre de voies	0
Fréquence de mise à jour maximum	40 mégahertz
Intervalle de mise à jour	1 à 65 535
Type de C N/A	12 bits double buffer
Limites de sorties (V) (Référence interne uniquement)	±5 dans une charge 50 Ω ±10 dans une charge indéterminée (impédance d'entrée haute)
Horloges de mise à jour	Horloge de mise à jour 1
Déclenchements	Sur front montant TTL, au connecteur d'entrée du déclenchement ou à la broche RTSI. Peut également être généré de façon interne par le logiciel
Bus de déclenchement RTSI	Oui
Sorties numériques	16 bits avec signal d'horloge
Groupement de formes d'ondes	0
Profondeur de mémoire de signal -Mode ARB -Mode Direct Digital Synthesis (DDS) (Synthèse numérique directe)	2 000 000 échantillons 16 bits (standard) 16 384 échantillons 16 bits maximum
Nombre maximal de sections de signaux	290
Nombre de buffers	1 à 1 000
Itérations de buffer	1 à 65 535
Décompte d'échantillons de buffer - Mode ARB -Mode DDS	256 échantillons minimum Profondeur de mémoire maximum Remarque : la taille du buffer doit être un multiple de 8 échantillons. Doit être égal à 16 384 échantillons. Si vous chargez moins d'échantillons, vous remarquerez alors que le contenu des sections de mémoire vides apparaissent également dans la génération de formes d'ondes
Sortie de repère	Niveau TTL, une sortie disponible pour chaque section
Taille de l'accumulateur DDS	32 bits

Tableau B-16. Caractéristiques des sorties analogiques et numériques—Périphériques Série 54XX (Suite)

Caractéristiques	AT-5411, PCI-5411
Fréquence de sortie maximum	16 mégahertz
Résolution de la fréquence de sortie (Mode DDS uniquement)	9,31 mégahertz
Atténuation de sortie (après le C N/A)	de 0 à 74 000 dB (Décibels) pour des sections de 0,001 dB
Facteur d'utilisation de sortie SYNC (% haut)	Niveau TTL, 20% à 80%
Horloge de référence PLL	1 mégahertz, 10 mégahertz ou 20 mégahertz
Sortie activée	Permutable par le logiciel sur MARCHE ou ARRET
Sortie activée	50Ω ou 75Ω (vidéo), logiciel pouvant être sélectionné
Filtre passe-bas	16 mégahertz, permutable par logiciel sur MARCHE ou ARRET
Filtre d'interpolation demi-bande numérique	80 MSPS, permutable par le logiciel sur MARCHE ou ARRET
Modes de fonctionnement de déclenchement	Unique, continu, échelonné ou en rafale



Remarque *Reportez-vous à votre manuel de référence du matériel pour connaître la configuration par défaut de votre périphérique.*

Tableau B-17. Caractéristiques des compteurs/timers—Périphériques Série 1200/Lab et portables

Périphérique	Circuit de compteur utilisé	Nombre de compteurs généraux disponibles	Base de temps disponible	Nombre de bits	Modes de fenêtrage disponibles	Sorties disponibles	Modes de sortie disponibles	Direction de comptage
Lab-NB Lab-LC Lab-PC+ SCXI-1200 DAQCard-1200 DAQPad-1200 PC-LPM-16 PCI-1200	8253	3 (2 avec entrée SOURCE au connecteur d'E/S)	Interne : 1 mégahertz; (PC-LPM-16: uniquement sur CTRB0) externe	16	Niveau haut ou front montant en fonction du mode de sortie	3	Reportez-vous au VI "ICTR-Contrôler" (ICTR Control.vi) pour la description des modes au chapitre 28, <i>Vis Compteurs avancés</i> .	Descendante
DAQCard-500 DAQCard 516 DAQCard-700 PC-516	8254	3 (2 avec entrée SOURCE au connecteur d'E/S)	Interne : 1 mégahertz uniquement sur CTRB0 ; externe	16	Niveau haut ou front montant en fonction du mode de sortie	3 (2 pour DAQCard-500)	Reportez-vous au VI "ICTR-Contrôler" (ICTR Control.vi) pour la description des modes au chapitre 28, <i>Vis Compteurs avancés</i> .	Descendante

Capacités logicielles des modules SCXI

Tableau B-18. Caractéristiques des entrées analogiques—Modules SCXI (1^{ère} partie)

Module	Nbre de voies	Gamme de tension d'entrée (V)	Gains ¹	Filtre ¹	Voies d'excitation ¹	Support de mode
SCXI-1100	32 DI	±10	1, 2, 5, 10, 20, 50, 100, 200, 500, 1000, 2000 (Logiciel/M) ¹	Filtre passe-bas (ou aucun filtre) avec une fréquence de coupure de 10 kilohertz ou 4 Hz (JS/M) ¹	—	Multiplexé
SCXI-1102	32 DI	±10	1, 100 (Logiciel/C) ¹	Passe bas de 1 Hz sur chaque voie	—	Multiplexé
SCXI-1120 SCXI-1121	8 DI (SCXI-1120) 4 DI (SCXI-1121)	±5	1, 2, 5, 10, 20, 50, 100, 200, 500, 1000, 2000 (JS/C) ¹	Filtre passe-bas avec une fréquence de coupure de 10 kilohertz ou 4 Hz (JS/M) ¹	SCXI-1121 uniquement : 4 JS/C ¹ (voies) de tension ou de courant d'excitation	Multiplexé ou parallèle
SCXI-1120D	8 DI (SCXI-1120) 4 DI (SCXI-1121)	±5	0.5, 1, 2.5, 5, 10, 25, 50, 100, 250, 500, 1000	4500, 24500 Hz	SCXI-1121 uniquement : 4 JS/C ¹ (voies) de tension ou de courant d'excitation	Multiplexé ou parallèle
SCXI-1122	16 DI ou 8 DI et 8 voies d'excitation Logiciel/M1	±10	0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50, 100, 200, 500, 1000, 2000 (Logiciel/M) ¹	Filtre passe-bas avec fréquence de coupure de 4 kilohertz ou 4 Hz	8 voies de tension ou de courant d'excitation en mode de balayage 4 fils	Multiplexé
SCXI-1140	8 DI, échantillonner-bloquer	±10	1, 10, 100, 200, 500 (DS/C) ¹	Aucun	—	Multiplexé ou parallèle

Tableau B-18. Caractéristiques des entrées analogiques—Modules SCXI (1^{ère} partie) (Suite)

Module	Nbre de voies	Gamme de tension d'entrée (V)	Gains ¹	Filtre ¹	Voies d'excitation ¹	Support de mode
SCXI-1141	8 DI	±5	1, 2, 5, 10, 20, 50, 100 (Logiciel/C) ¹	Filtre passe-bas avec fréquence ² de coupure de 10 Hz à 25 kilohertz (Logiciel/M) ¹ (désactivé en fonction de chaque voie)	—	Multiplexé ou parallèle

¹DS/C = sélectionnable par interrupteur et par voie, JS/C = sélectionnable par cavalier et par voie, JS/M = sélectionnable par cavalier et par module, Logiciel/C = sélectionnable par logiciel et par voie, Logiciel/M = sélectionnable par logiciel et par module

²Le filtre du module SCXI-1141 est réglé automatiquement. LabVIEW définit la fréquence du filtre en fonction des fréquences de balayage utilisées avec le module.

Tableau B-19. Caractéristiques des sorties analogiques—Modules SCXI

Module	Nombre de voies	Gamme de tension de sortie (V ou mA)	Support de mode
SCXI-1124	6 tensions ou courants	0 à 1, 0 à 5, 0 à 10, ±1, ±5, ±10 (sélectionnable par logiciel) ou 0 à 20 mA	Multiplexé

Tableau B-20. Caractéristiques des relais—Modules SCXI

Module	Nbre de voies ¹	Armé ou non armé	Position de démarrage des relais ¹	Support de mode
SCXI-1160	16	Armé	Laisse les relais en position lors de la coupure d'alimentation	Multiplexé
SCXI-1161	8	Non armé	Positionne sur Normally Closed (NC) (Fermeture normale), lors de la remise à zéro du module	Multiplexé

¹Vous pouvez définir ou redéfinir chaque relais du module SCXI de façon individuelle sans affecter d'autres relais ou vous pouvez modifier tous les relais à la fois.

Tableau B-21. Caractéristiques des entrées et des sorties numériques—Modules SCXI

Module	Type de module	Nombre de voies ¹	Gamme de tension d'entrée	Support de mode
SCXI-1162	Entrée	32 (isolation optique)	0 à 5 V	Support parallèle; lorsque connecté à un périphérique DIO-24, DIO-96 ou DIO-32F Support multiplexé avec tout périphérique supportant le SCXI.
SCXI-1162HV	Entrée	32 (isolation optique)	Signaux AC ou DC jusqu'à ± 240 V	Support parallèle; lorsque connecté à un périphérique DIO-24, DIO-96 ou DIO-32F Support multiplexé avec tout périphérique supportant le SCXI.
SCXI-1163	Sortie	32 (isolation optique)	0 à 5 V	Support parallèle; lorsque connecté à un périphérique DIO-24, DIO-96 ou DIO-32F Support multiplexé avec tout périphérique supportant le SCXI.
SCXI-1163R*	Sortie	32 (isolation optique)	± 240 V	Support parallèle; lorsque connecté à un périphérique DIO-24, DIO-96 ou DIO-32F Support multiplexé avec tout périphérique supportant le SCXI.

¹ La fonctionnalité est équivalente au module SCXI-1163 mais elle incorpore des relais statiques au lieu de sorties numériques.

Tableau B-22. Guide de sélection des blocs de terminaison—Modules SCXI

Module SCXI	Blocs de terminaison	Capteur de compensation de soudure froide (CJC)
SCXI-1100 SCXI-1102	SCXI-1303 SCXI-1300	Thermistance Capteur IC
SCXI-1120 SCXI-1121	SCXI-1320 SCXI-1321 ¹ SCXI-1327 SCXI-1328	Capteur IC Capteur IC Thermistance Thermistance
SCXI-1122	SCXI-1322	Thermistance
SCXI-1124	SCXI-1325	—
SCXI-1140	SCXI-1301 SCXI-1304	— —
SCXI-1141	SCXI-1304	—
SCXI-1160	SCXI-1324	—
SCXI-1161	Terminaisons sans bornier à vis situées dans le module	—
SCXI-1162 SCXI-1162HV SCXI-1163 SCXI-1163R	SCXI-1326	—
SCXI-1180	SCXI-1302	—
SCXI-1181	SCXI-1300 SCXI-1301	Capteur IC —
SCXI-1200	SCXI-1302 CB-50	— —

¹ SCXI-1121 uniquement

Tableau B-23. Programmabilité de la configuration des entrées analogiques

Périphérique	Gain	Couplage
Périphériques 5102	Par voie	Par voie

Tableau B-24. Programmabilité de la configuration des entrées analogiques

Périphérique	Nbre de voies	Résolution	Gains	Gamme (V)	Entrée FIFO (mots)	Balayage
Périphériques 5102	2	8 bits	1, 5, 20, 100	± 5	663546	1 ou 2 voies dans n'importe quel ordre sans répétition

**Remarque**

Par périphérique signifie que vous pouvez sélectionner la valeur d'un paramètre avec des cavaliers matériels et que la sélection affecte tous les groupes de voies du périphérique. Par groupe signifie que vous pouvez programmer la sélection via le logiciel et que celle-ci affecte toutes les voies utilisées au même moment. Par voie signifie que vous pouvez programmer la sélection avec des cavaliers matériels ou via le logiciel voie par voie. Lorsque la valeur spécifique d'un paramètre est affichée, elle est enregistrée.

Capacités matérielles des périphériques de sorties analogiques dédiés

Tableau B-25. Caractéristiques des sorties analogiques—Périphériques de sorties analogiques dédiés

Périphérique	Nbre de voies	Type de C N/A	Limites de sortie	Horloges de mise à jour	Groupement de formes d'ondes	Méthode de transfert
AT-AO-6 AT-AO-10 NB-AO-6	0 à 5, 6 à 9*	12 bits double buffer avec FIFO de 1 K pour des voies d'horloge de mise à jour 1	$\pm 10V$, $\pm V_{réf1}$, 0 à 10 V, 0 à $V_{réf1}$, 4 à 20 mA	L'horloge de mise à jour 1 est ctr 0 ou une mise à jour externe. Les voies de l'horloge de mise à jour 1 sont 0, 1, 2, 3, 4, 5, 6*, 7*, 8*, 9*, 0 à 1, 0 à 3, 0 à 5, 0 à 7*, 0 à 9*. L'horloge de mise à jour 2 est ctr1. Les voies de l'horloge 2 sont 2, 3, 4, 5, 6*, 7*, 8*, 9*, 2 à 3, 2 à 5, 2 à 7*, 2 à 9*; la gamme des signaux de base de temps est 1 000 000, 100 000, 10 000, 1 000, 100	Pour l'horloge de mise à jour 1 les voies sont n'importe quelle voie N ou un ensemble de paires de voies : $0-N$; pour l'horloge de mise à jour 2 les voies sont $2-N$, règles identiques à ce qui précède : $N\#6, N\#10^*$	Voies de l'horloge de mise à jour 1 : DMA, interruptions ;voies de l'horloge de mise à jour 2 : interruptions
PC-AO-2DC (Plug and Play)	0, 1	—	0 à 10V, $\pm 5V$, 0–20 mA sélectionnable par logiciel	—	—	—

Tableau B-25. Caractéristiques des sorties analogiques—Périphériques de sorties analogiques dédiés (Suite)

Périphérique	Nbre de voies	Type de C N/A	Limites de sortie	Horloges de mise à jour	Groupe ment de formes d'ondes	Méthode de transfert
DAQCard-AO-2DC	0, 1	—	0 à 10V, ± 5 V, 0–20mA sélectionnable par logiciel	—	—	—
*AT-AO-10 uniquement						

Capacités matérielles des périphériques d'acquisition de signaux dynamiques

Tableau B-26. Programmabilité de la configuration des entrées analogiques—Périphériques d'acquisition de signaux dynamiques

Périphérique	Gain	Gamme (V)	Polarité	Asym/DIFF	Couplage
EISA-A2000 NB-A2000	1	± 5	Bipolaire	Asym	Par voie
NB-A2100 AT-DSP2200	1	± 2.828	Bipolaire	Asym	Par groupe
NB-A2150 AT-A2150	1	± 2.828	Bipolaire	Asym	Par paire de voies 0 et 1, 2 et 3

**Remarque**

Par périphérique signifie que vous pouvez sélectionner la valeur d'un paramètre avec des cavaliers matériels et que la sélection affecte tous les groupes de voies du périphérique. Par groupe signifie que vous pouvez programmer la sélection via le logiciel et que celle-ci affecte toutes les voies utilisées au même moment. Par voie signifie que vous pouvez programmer la sélection avec des cavaliers matériels ou via le logiciel voie par voie. Lorsque la valeur spécifique d'un paramètre est affichée, elle est enregistrée.

Tableau B-27. Caractéristiques des sorties analogiques—Périphériques d'acquisition de signaux

Périphérique	Nbre de voies	Type de C N/A	Limites de sortie	Horloges de mise à jour	Méthode de transfert
PCI-4451 PCI-4552	0, 1	18 bits	± 10 V, ± 1 V, ± 100	Horloge de mise à jour 1	DMA

Tableau B-28. Caractéristiques des entrées analogiques—Périphériques d'acquisition de signaux dynamiques

Périphérique	Nombre de voies	Résolution	Gamme (V)	Entrée FIFO (mots)	Déclenchement	Balayage	Fréquence d'échantillonnage max (E/s)	Méthode de transfert
EISA-A2000 NB-A2000	4 Asym	12 bits	±5	EISA :51; NB :1,024	Déclenchement logiciel, pré-déclenchement et post-déclenchement avec retard de post-déclenchement et de déclenchement numérique ou analogique	0, 1, 2, 3, 0 et 1, 2 et 3, 0 à 3.	1M	DMA, interruptions
NB-A2100 NB-A2150	2 Asym	16 bits	±2,828	32	Déclenchement logiciel, pré-déclenchement et post-déclenchement avec déclenchement et numérique ou analogique	A2150 : 0, 1, 2, 3, 0 et 1, 2 et 3, 0 à 3 ; A2100 : 0, 1, 0 et 1	2100 :48 k, 2150 :24 k, 2150C : 48 k, 2150S : 51,2 k	DMA, interruptions
AT-A2150	4 Asym	16 bits	±2,828	—	Déclenchement logiciel, pré-déclenchement et post-déclenchement avec déclenchement numérique ou analogique	0, 1, 2, 3, 0 et 1, 2 et 3, 0 et 3	2150 :24 k 2150 :51,2k	DMA, interruptions

Capacités matérielles des périphériques numériques dédiés

Tableau B-29. Capacités matérielles numériques—Périphériques d'E/S numériques

Périphérique	Type de port	Nombres des ports	Modes de handshaking	Direction	Horloges DIO	Méthode de transfert
AT-DIO-32F NB-DIO-32F	Port 8 bits	0, 1, 2, 3	Port 8 bits Handshaking activé ou désactivé ; modes de handshaking extensifs	Lecture ou écriture	Deux hor- loges 16 bits dis- ponibles avec base de temps variable	DMA pour chaque groupe ; double DMA pour les groupes contenant le port 0
	Port 2 bits	4	Aucun handshaking	Lecture ou écriture	Aucune	Interrogation logicielle
PC-DIO-24 NB-DIO-24 DAQCard-DIO-24	Port 8 bits	0, 1	Handshaking activé ou désactivé	Lecture ou écriture, le port 0 est peut-être bidirectionnel	Aucune	Interruptions
	Port 8 bits	2	Aucun handshaking ; inutilisable si le port 0 ou 1 utilise le handshaking	Lecture ou écriture	Aucune	Interrogation logicielle
PC-DIO-96 PCI-DIO-96 NB-DIO-96 DAQPad-6507	Port 8 bits	0, 1, 3, 4, 6, 7, 9, 10	Handshaking activé ou désactivé	Lecture ou écriture, les ports 0, 3, 6, et 9 peuvent être bidirectionnels	Aucune	Interruptions
	Port 8 bits	2, 5, 8, 11	Aucun handshaking ; inutilisable si les ports A et B du circuit 8255 utilisent le handshaking	Lecture ou écriture	Aucune	Interrogation logicielle
PC-OPDIO-16 (Plug and Play)	Port 8 bits à isolation optique	0, 1	—	Port 0 pour la sortie (écriture) ; port 1 pour l'entrée (lecture)	Aucune	E/S programmées

Capacités matérielles des périphériques de séquençement dédiés

Tableau B-30. Capacités matérielles numériques—Périphériques de séquençement dédiés

Périphérique	Type de port	Numéros des ports	Modes de Handshaking	Direction	Horloges DIO	Méthode de transfert
PC-TIO-10 NB-TIO-10	Ports 8 bits	0, 1	Aucun handshaking	Contrôle de la direction en représentation binaire	Aucune	Interrogation logicielle

Tableau B-31. Caractéristiques des compteurs/timers—Périphériques de séquençement dédiés

Périphérique	Circuit de compteur utilisé	Nombre de compteurs généraux disponibles	Bases de temps disponibles	Nombre de bits	Modes de fenêtrage disponibles	Sorties disponibles	Modes de sortie disponibles	Direction du comptage
PC-TIO-10 NB-TIO-10	Am-9513	10 (8 ont des entrées SOURCE au connecteur d'E/S)	Interne : 5 mégahertz (uniquement sur CTR5 et CTR10), 1 mégahertz, 100 kilohertz, 10 kilohertz, 1 kilohertz, 100 hertz ; externe	16	niveau haut, niveau bas, front montant, front descendant	10	Impulsion TC, bascule TC	Haut ou bas

Capacités matérielles des périphériques 5102

Tableau B-32. Programmabilité de la configuration des entrées analogiques

Périphérique	Gain	Couplage
Périphériques 5102	Par voie	Par voies

Tableau B-33. Caractéristiques des entrées analogiques

Périphérique	Nbre de voies	Résolution	Gains	Gamme (V)	Entrée FIFO (mots)	Balayage
Périphériques 5102	2	8 bits	1, 5, 20, 100	±5	663 000	1 ou 2 voies dans n'importe quel ordre sans répétition

Tableau B-34. Caractéristiques des entrées analogiques, (2^{ème} partie)

Périphérique	Déclenchements	Fréquence d'échantillonnage maximum (E/s)
Périphériques 5102	Logiciel, Pré, Post, Analogique	20 000 000 temps réel



Messages d'interface multiligne GPIB

Cette annexe liste les messages d'interface multiligne qui sont des commandes définies par IEEE 488. Les messages d'interface multiligne gèrent le GPIB : ils effectuent des tâches telles que l'initialisation du bus, l'adressage et le désadressage des périphériques et la définition des modes des périphériques pour la programmation locale ou à distance. Ces messages sont envoyés et reçus avec ATN TRUE. La liste qui suit inclut les caractères mnémoniques et les messages qui correspondent aux fonctions d'interface.

Pour plus d'informations concernant ces messages, reportez-vous au ANSI/IEEE Standard 488.1-1987, *IEEE Standard Digital Interface for Programmable Instrumentation*.

Messages d'interface multiligne

Hex	Oct	Déc	ASCII	Msg
00	000	0	NUL	—
01	001	1	SOH	GTL
02	002	2	STX	—
03	003	3	ETX	—
04	004	4	EOT	SDC
05	005	5	ENQ	PPC
06	006	6	ACK	—
07	007	7	BEL	—
08	010	8	BS	GET
09	011	9	HT	TCT

Hex	Oct	Déc	ASCII	Msg
0A	012	10	LF	—
0B	013	11	VT	—
0C	014	12	FF	—
0D	015	13	CR	—
0E	016	14	SO	—
0F	017	15	SI	—
10	020	16	DLE	—
11	021	17	DC1	LLO
12	022	18	DC2	—
13	023	19	DC3	—
14	024	20	DC4	DCL
15	025	21	NAK	PPU
16	026	22	SYN	—
17	027	23	ETB	—
18	030	24	CAN	SPE
19	031	25	EM	SPD
1A	032	26	SUB	—
1B	033	27	ESC	—
1C	034	28	FS	—
1D	035	29	GS	—
1E	036	30	RS	—
1F	037	31	US	—
20	040	32	SP	MLA0
21	041	33	!	MLA1
22	042	34	“	MLA2
23	043	35	#	MLA3

Hex	Oct	Déc	ASCII	Msg
24	044	36	\$	MLA4
25	045	37	%	MLA5
26	046	38	&	MLA6
27	047	39	'	MLA7
28	050	40	(MLA8
29	051	41)	MLA9
2A	052	42	*	MLA10
2B	053	43	+	MLA11
2C	054	44	,	MLA12
2D	055	45	-	MLA13
2E	056	46	.	MLA14
2F	057	47	/	MLA15
30	060	48	0	MLA16
31	061	49	1	MLA17
32	062	50	2	MLA18
33	063	51	3	MLA19
34	064	52	4	MLA20
35	065	53	5	MLA21
36	066	54	6	MLA22
37	067	55	7	MLA23
38	070	56	8	MLA24
39	071	57	9	MLA25
3A	072	58	:	MLA26
3B	073	59	;	MLA27
3C	074	60	<	MLA28
3D	075	61	=	MLA29

Hex	Oct	Déc	ASCII	Msg
3E	076	62	>	MLA30
3F	077	63	?	UNL
40	100	64	@	MTA0
41	101	65	A	MTA1
42	102	66	B	MTA2
43	103	67	C	MTA3
44	104	68	D	MTA4
45	105	69	E	MTA5
46	106	70	F	MTA6
47	107	71	G	MTA7
48	110	72	H	MTA8
49	111	73	I	MTA9
4A	112	74	J	MTA10
4B	113	75	K	MTA11
4C	114	76	L	MTA12
4D	115	77	M	MTA13
4E	116	78	N	MTA14
4F	117	79	O	MTA15
50	120	80	P	MTA16
51	121	81	Q	MTA17
52	122	82	R	MTA18
53	123	83	S	MTA19
54	124	84	T	MTA20
55	125	85	U	MTA21
56	126	86	V	MTA22
57	127	87	W	MTA23

Hex	Oct	Déc	ASCII	Msg
58	130	88	X	MTA24
59	131	89	Y	MTA25
5A	132	90	Z	MTA26
5B	133	91	[MTA27
5C	134	92	\	MTA28
5D	135	93]	MTA29
5E	136	94	^	MTA30
5F	137	95	_	UNT
60	140	96	`	MSA0,PPE
61	141	97	a	MSA1,PPE
62	142	98	b	MSA2,PPE
63	143	99	c	MSA3,PPE
64	144	100	d	MSA4,PPE
65	145	101	e	MSA5,PPE
66	146	102	f	MSA6,PPE
67	147	103	g	MSA7,PPE
68	150	104	h	MSA8,PPE
69	151	105	i	MSA9,PPE
6A	152	106	j	MSA10,PPE
6B	153	107	k	MSA11,PPE
6C	154	108	l	MSA12,PPE
6D	155	109	m	MSA13,PPE
6E	156	110	n	MSA14,PPE
6F	157	111	o	MSA15,PPE
70	160	112	p	MSA16,PPD
71	161	113	q	MSA17,PPD

Hex	Oct	Déc	ASCII	Msg
72	162	114	r	MSA18,PPD
73	163	115	s	MSA19,PPD
74	164	116	t	MSA20,PPD
75	165	117	u	MSA21,PPD
76	166	118	v	MSA22,PPD
77	167	119	w	MSA23,PPD
78	170	120	x	MSA24,PPD
79	171	121	y	MSA25,PPD
7A	172	122	z	MSA26,PPD
7B	173	123	{	MSA27,PPD
7C	174	124		MSA28,PPD
7D	175	125	}	MSA29,PPD
7E	176	126	~	MSA30,PPD
7F	177	127	DEL	—

Définitions des messages

Caractères mnémoniques	Définition
DCL	Réinitialiser l'appareil
GET	Grouper le déclenchement d'exécution
GTL	Mettre en local
LLO	Verrouillage local
MLA	Adresse de mon auditeur (listener)
MSA	Mon adresse secondaire
MTA	Adresse de mon transmetteur (talker)

Caractères mnémoniques	Définition
PPC	Configurer l'interrogation parallèle
PPD	Désactiver l'interrogation parallèle
PPE	Activer l'interrogation parallèle
PPU	Déconfigurer l'interrogation parallèle
SDC	Réinitialiser l'appareil sélectionné
SPD	Désactiver l'interrogation en série
SPE	Activer l'interrogation en série
TCT	Prendre le contrôle
UNL	Unlisten
UNT	Untalk

Informations à l'attention du client

Cette annexe contient les formulaires qui vous permettront de réunir les renseignements nécessaires pour vous aider à résoudre vos problèmes techniques, ainsi qu'une fiche que vous pouvez utiliser pour nous signaler vos remarques sur nos documentations de produits. Lorsque vous nous contactez, veuillez nous fournir les renseignements contenus dans le formulaire de support technique et, le cas échéant, dans la fiche de configuration de votre système de façon à ce que nous puissions répondre à vos questions dans les plus brefs délais.

National Instruments offre une assistance technique par téléphone, télécopie et par l'intermédiaire de systèmes électroniques pour que vous puissiez obtenir rapidement l'information que vous souhaitez. Nos services de support électroniques comprennent un BBS, un site FTP, un système Fax-on-demand et un service de courrier électronique. Si vous rencontrez des problèmes de matériel ou de logiciel, essayez tout d'abord les services de support électroniques. Si les renseignements fournis par ces systèmes sont insuffisants, nos centres de support technique pourvus d'ingénieurs d'application, offrent également un support téléphonique et par télécopie.

Services électroniques

Support par site FTP

Pour accéder à notre site FTP, ouvrez une session sur l'hôte Internet, `ftp.natinst.com`, comme anonyme et utilisez votre adresse Internet, par exemple `pauldupont@ailleurs.com`, comme mot de passe. Les fichiers et documents de support se trouvent dans les répertoires `/support`.

Support par le système Fax-on-Demand

Le système Fax-on-Demand est un système de recherche documentaire accessible 24 heures sur 24 contenant une bibliothèque documentaire portant sur une vaste gamme d'informations techniques. Vous pouvez accéder au système Fax-on-Demand avec un téléphone à touches en composant le 512 418 1111.

Support par courrier électronique (disponible aux Etats-Unis seulement)

Vous pouvez poser vos questions techniques à l'équipe d'ingénieurs d'applications par courrier électronique à l'adresse Internet indiquée ci-dessous. N'oubliez pas de faire figurer vos nom, adresse et numéro de téléphone, de façon à ce que nous puissions vous communiquer des solutions et suggestions.

`support@natinst.com`

Support par affichage électronique

National Instruments dispose de sites BBS et FTP spécialisés en support technique accessible 24 heures sur 24, contenant un large éventail de fichiers et de documents pour répondre aux questions les plus courantes de ses clients. Ces sites vous permettent également de télécharger les tout derniers drivers d'instruments, mises à jour et programmes d'exemple. Pour obtenir des instructions enregistrées sur la façon d'utiliser les services BBS et FTP, et pour obtenir les services d'informations automatisés BBS, composez le 512 795 6990. Vous pouvez accéder à ces services de la manière suivante :

En France : 01 48 65 15 59

Jusqu'à 9,600 bauds, 8 bits de données, 1 bit d'arrêt, aucune parité

Aux Etats-Unis : 512 794 5422

Jusqu'à 14,400 bauds, 8 bits de données, 1 bit d'arrêt, aucune parité

Au Royaume-Uni : 01635 551422

Jusqu'à 9,600 bauds, 8 bits de données, 1 bit d'arrêt, aucune parité

Support par téléphone et télécopie

National Instruments dispose de bureaux répartis dans le monde entier. Utilisez la liste suivante pour localiser le numéro de support technique de votre pays. Si National Instruments n'a pas de bureau de représentation dans votre pays, contactez le revendeur de votre logiciel pour obtenir de l'aide.

Pays	Téléphone	Télécopie
Allemagne	089 741 31 30	089 714 60 35
Australie	03 9879 5166	03 9879 6277
Autriche	0662 45 79 90 0	0662 45 79 90 19
Belgique	02 757 00 20	02 757 03 11
Bésil	011 288 3336	011 288 8528
Canada (Ontario)	905 785 0085	905 785 0086
Canada (Québec)	514 694 8521	514 694 4399
Corée	02 596 7456	02 596 7455
Danemark	45 76 26 00	45 76 26 02
Espagne	91 640 0085	91 640 0533
Etats-Unis	512 795 8248	512 794 5678
Finlande	09 725 725 11	09 725 725 55
France	01 48 14 24 24	01 48 14 24 14
Hong Kong	2645 3186	2686 8505
Israël	03 6120092	03 6120095
Italie	02 413091	02 41309215
Japon	03 5472 2970	03 5472 2977
Mexique	5 520 2635	5 520 3282
Norvège	32 84 84 00	32 84 86 00
Pays-bas	0348 433466	0348 430673
Royaume-Uni	01635 523545	01635 523154
Singapour	2265886	2265887
Suède	08 730 49 70	08 730 43 70
Suisse	056 200 51 51	056 200 51 55
Taiwan	02 377 1200	02 737 4644

Formulaire de support technique

Photocopiez ce formulaire et mettez-le à jour chaque fois que vous faites des changements d'ordre logiciel ou matériel, et utilisez la copie de ce formulaire complété comme référence pour votre configuration actuelle. Complétez ce formulaire avec soin avant de contacter National Instruments pour obtenir le support technique nécessaire, afin de permettre à nos ingénieurs d'application de répondre plus efficacement à vos questions.

Si vous utilisez des produits logiciels et matériels de National Instruments ayant rapport au problème, ajoutez les fiches de configuration se trouvant dans les manuels de l'utilisateur correspondants. Ajoutez toute page supplémentaire si nécessaire.

Nom _____

Société _____

Adresse _____

Téléphone (___) _____ Télécopie (___) _____

Marque d'ordinateur _____ Modèle _____ Processeur _____

Système d'exploitation (inclure le numéro de version) _____

Fréquence d'horloge _____ MHz RAM _____ Mo Carte graphique _____

Souris ___oui ___non Autres cartes installées _____

Capacité du disque dur _____ Mo Marque _____

Instruments utilisés _____

Modèle de produit matériel de National Instruments _____ Révision _____

Configuration _____

Produit logiciel de National Instruments _____ Version _____

Configuration _____

Description du problème : _____

Listez tout message d'erreur : _____

Le problème apparaît chaque fois que les opérations suivantes sont effectuées : _____

Fiche de configuration logicielle et matérielle LabVIEW

Inscrivez les paramètres et les modifications apportées à votre matériel et à votre logiciel. Complétez une nouvelle copie de ce formulaire chaque fois que vous modifiez la configuration de votre logiciel ou de votre matériel, et utilisez ce formulaire comme référence pour votre configuration actuelle. Complétez ce formulaire avec soin avant de contacter National Instruments pour obtenir un support technique afin de permettre à nos ingénieurs d'application de répondre plus efficacement à vos questions.

Produits de National Instruments

Matériel DAQ _____

Niveau d'interruption du matériel (IRQ) _____

Canaux DMA du matériel _____

Adresse de base d'E/S du matériel _____

Choix de programmation _____

Version NI-DAQ ou LabVIEW _____

Autres cartes dans le système _____

Adresse de base d'E/S des autres cartes _____

Canaux DMA des autres cartes _____

Niveau d'interruption des autres cartes _____

Autres produits

Marque et modèle de l'ordinateur _____

Microprocesseur _____

Fréquence ou vitesse d'horloge _____

Type de carte vidéo installée _____

Version du système d'exploitation _____

Mode du système d'exploitation _____

Langage de programmation _____

Version du langage de programmation _____

Autres cartes présentes dans le système _____

Adresse de base d'E/S des autres cartes _____

Canaux DMA des autres cartes _____

Niveau d'interruption des autres cartes _____

Vos commentaires sur la documentation...

National Instruments vous encourage à nous signaler vos remarques sur les documentations livrées avec nos produits. Ces informations nous aideront à vous garantir la qualité que vous attendez de nos produits.

Titre : *Manuel de référence des VIs et des fonctions de LabVIEW™*

Date d'édition : Juillet 1998

Référence : 321981A-01

Veillez apporter des commentaires sur l'exhaustivité, la clarté et l'organisation de ce manuel :

Si vous avez trouvé des erreurs, indiquez les numéros de pages correspondants et décrivez la nature de ces erreurs :

Nous vous remercions de votre contribution.

Nom _____

Fonction _____

Société _____

Adresse _____

Adresse électronique _____

Téléphone (____) _____ Télécopie (____) _____

A envoyer à : National Instruments
Centre d'Affaires Paris-Nord
Immeuble "Le Continental"
BP 217
93153 Le Blanc-Mesnil Cedex

A télécopier au : numéro de fax de
National Instruments France
01 48 14 24 14

Index

Nombres

1200-Etalonner (1200 Calibrate.vi), 29-2

A

A partir d'un format exponentiel/fract/ing, 6-19

A partir d'un format hexa, 6-20

A partir d'un format octal, 6-20

A x B, 45-2

A x B Complexe (Complex A x B.vi), 45-3

A x Vecteur (A x Vector.vi), 45-2

A x vecteur complexe (Complex A x Vector.vi), 45-4

A2000-Configurer (A2000 Config.vi), 29-3

A2000-Etalonner (A2000 Calibrate.vi), 29-2

A2100-Configurer (A2100 Config.vi), 29-4

A2100-Etalonner (A2100 Calibrate.vi), 29-4

A2150-Configurer (A2150 Config.vi), 29-5

A2150-Etalonner (A2150 Calibrate.vi) (Macintosh), 29-6

Acquérir un sémaphore (Acquire Semaphore.vi), 13-19

Activer le repérage de menus, 12-9

Adapter le type, 13-7

AECreat Comp Descriptor, 52-15

AECreat Descriptor List, 52-16

AECreat Logical Descriptor, 52-15

AECreat Object Specifier, 52-16

AECreat Range Descriptor, 52-16

AECreat Record, 52-17

AESend, 52-14

AESend Abort, 52-8

AESend Close, 52-8

AESend Do Script, 52-6

AESend Finder Open, 52-2, 52-6

AESend Open, 52-7

AESend Open Document, 52-7

AESend Open, Run, Close, 52-8

AESend Print Document, 52-7

AESend Quit Application, 52-7

AESend Run, 52-9

AESend VI Active?, 52-9

AI-Acquérir des signaux (AI Acquire Waveforms.vi), 15-2

AI-Acquérir un signal (AI Acquire Waveform.vi), 15-2

AI-Balayage simple (AI Single Scan.vi) (intermédiaire), 16-4

AI-BalayageUnique (AI SingleScan.vi) (avancé), 18-15

AI-Balayer en continu (AI Continuous Scan.vi), 17-2

AI-Balayer un signal (AI Waveform Scan.vi), 17-4

AI-Configurer (AI Config.vi), 16-3

AI-Configurer un buffer (AI Buffer Config.vi), 18-2

AI-Configurer un déclenchement (AI Trigger Config.vi), 18-16

AI-Configurer un groupe (AI Group Config.vi), 18-7

AI-Configurer un matériel (AI Hardware Config.vi), 18-9

AI-Configurer une horloge (AI Clock Config.vi), 18-4

AI-Contrôler (AI Control.vi), 18-6

AI-Démarrer (AI Start.vi), 16-5

AI-Echantillonner des voies (AI Sample Channels.vi), 15-3

AI-Echantillonner une voie (AI Sample Channel.vi), 15-3

AI-Lire (AI Read.vi), 16-3

AI-Lire un balayage (AI Read One Scan.vi), 17-3

AI-Lire un buffer (AI Buffer Read.vi), 18-3

- AI-Paramètre (AI Parameter.vi), 18-14
- AI-Réinitialiser (AI Clear.vi), 16-2
- Ajustement de Lev-Mar non linéaire (Nonlinear Lev-Mar Fit.vi), 43-5
- Ajustement exponentiel (Exponential Fit.vi), 43-2
- Ajustement linéaire (Linear Fit.vi), 43-4
- Ajustement linéaire général (moindre carré) (General LS Linear Fit.vi), 43-3
- Ajustement polynomial général (General Polynomial Fit.vi), 43-3
- Ajuster la constante de mise à l'échelle (Scaling Constant Tuner.vi), 29-18, 30-9
- AllSPoll, 35-4
- Analyseur harmonique (Harmonic Analyzer.vi), 40-4
- Annuler une notification (Cancel Notifier.vi), 13-10
- ANOVA 1D (1D ANOVA.vi), 44-2
- ANOVA 2D (2D ANOVA.vi), 44-3
- ANOVA 3D (3D ANOVA.vi), 44-4
- AO-6/10-Etalonner (AO-6/10 Calibrate.vi) (Windows), 29-6
- AO-Attendre (AO Wait.vi), 20-3
- AO-Configurer (AO Config.vi), 20-2
- AO-Configurer un buffer (AO Buffer Config.vi), 22-2
- AO-Configurer un déclenchement et un fenêtrage (AO Trigger and Gate Config.vi) (Windows), 22-5
- AO-Configurer un groupe (AO Group Config.vi), 22-4
- AO-Configurer un matériel (AO Hardware Config.vi), 22-4
- AO-Configurer une horloge (AO Clock Config.vi), 22-3
- AO-Contrôler (AO Control.vi), 22-3
- AO-Démarrer (AO Start.vi), 20-3
- AO-Ecrire (AO Write.vi), 20-3
- AO-Ecrire dans un buffer (AO Buffer Write.vi), 22-2
- AO-Ecrire une mise à jour (AO Write One Update.vi), 21-5
- AO-Générer des signaux (AO Generate Waveforms.vi), 19-2
- AO-Générer des signaux (AO Waveform Gen.vi), 21-4
- AO-Générer en continu (AO Continuous Gen.vi), 21-2
- AO-Générer un signal (AO Generate Waveform.vi), 19-2
- AO-Mettre à jour des voies (AO Update channels.vi), 19-3
- AO-Mettre à jour une seule fois (AO Single Update.vi), 22-5
- AO-Mettre à jour une voie (AO Update channel.vi), 19-2
- AO-Paramètre (AO Parameter.vi), 22-4
- AO-Réinitialiser (AO Clear.vi), 20-2
- Aplatir en chaîne de caractères, 13-4
- Arrêter, 12-6
- Arrêter un compteur (Counter Stop.vi), 27-4
- Assembler, 8-4
- Assembler par nom, 8-5
- Attendre (ms), 10-11
- Attendre un multiple de ms, 10-11
- Attendre une notification (Wait on Notification.vi), 13-12
- Attendre une notification multiple (Wait on Notification from Multiple.vi), 13-12
- Attendre une occurrence, 13-22
- Attendre+ (ms) (Wait+ (ms).vi), 27-8
- Au format décimal, 6-20
- Au format exponentiel, 6-21
- Au format fractionnel, 6-21
- Au format hexadécimal, 6-21
- Au format ingénieur, 6-20
- Au format octal, 6-21
- Autocorrélation (AutoCorrelation.vi), 39-2

B

Balayer une chaîne de caractères, 6-13
 Bip (Beep.vi), 13-2
 boîte de calcul, 3-3
 Boîte de dialogue de fichier, 11-18
 Boîte de dialogue deux boutons, 10-11
 Boîte de dialogue un bouton, 10-10
 Booléen en (0,1), 5-3
 boucle For, 3-2
 boucle While, 3-3
 Bruit aléatoire périodique (Periodic Random Noise.vi), 38-4
 Bruit blanc gaussien (Gaussian White Noise.vi), 38-3
 Bruit blanc uniforme (Uniform White Noise.vi), 38-10
 Bytes at Serial Port, 36-1

C

cac – Devenir le contrôleur actif, 34-9
 Cartésien vers polaire 1D (1D Rectangular To Polar.vi), 46-3
 Chaîne d'appel, 12-3
 Chaîne de caractères au format tableur en tableau, 6-16
 Chaîne de caractères en chemin, 6-23, 11-22
 Chaîne de caractères en IP, 48-3
 Chaîne de caractères en tableau d'octets, 4-12, 6-23
 Chaîne de caractères vide, 6-24
 Changer les bases d'unité, 4-11
 Chemin en chaîne de caractères, 11-21
 Chemin en tableau de chaînes de caractères, 6-23, 11-21
 Chemin en une chaîne de caractères, 6-23
 Chemin vide, 11-30
 Chemin/Chaîne de caractères vide ?, 9-6
 Chiffre décimal ?, 9-6
 Chiffre hexa ?, 9-8

Chiffre octal ?, 9-10
 Classe lexicale, 9-9
 Classer un tableau 1D, 7-8
 Close All PPC Ports, 53-2
 Cluster en tableau, 7-5, 8-5
 cmd – Envoyer des commandes IEEE 488, 34-10
 Code Interface Node, 13-2
 Coefficients ajustement linéaire (Linear Fit Coefficients.vi), 43-4
 Coefficients bande étroite RIF (FIR Narrowband Coefficients.vi), 41-7
 Coefficients cascades->directs (Cascade->Direct Coefficients.vi), 41-3
 Coefficients d'ajustement exponentiel (Exponential Fit Coefficients.vi), 43-2
 Coefficients de Bessel (Bessel Coefficients.vi), 41-2
 Coefficients de Butterworth (Butterworth Coefficients.vi), 41-2
 Coefficients de Chebyshev (Chebyshev Coefficients.vi), 41-4
 Coefficients de Chebyshev inv. (Inv Chebyshev Coefficients.vi), 41-12
 Coefficients elliptiques (Elliptic Coefficients.vi), 41-5
 Coefficients fenêtrés RIF (FIR Windowed Coefficients.vi), 41-10
 Comblé avec des zéros (Zero Padder.vi), 39-20
 Comparaison booléenne, 9-2
 Comparaison de caractères, 9-4
 Comparaison de chaînes de caractères, 9-2
 Comparaison de clusters, 9-2
 Comparaison numérique, 9-2
 Complexe en polaire, 4-21
 Complexe en Re/Im, 4-21
 Compter des événements ou un temps (Count Events or Time.vi), 26-2
 Compteur d'impulsions d'horloge (ms), 10-10
 Compteurs adjacents (Adjacent Counters.vi), 27-2

- Concaténer des chaînes de caractères, 6-7
- Config. maître/esclave (Master Slave Config.vi), 29-16
- Configurer des occurrences DAQ (DAQ Occurrence Config.vi) (Windows), 29-10
- Configurer un buffer numérique (Digital Buffer Config.vi), 25-3
- Configurer un compteur descendant ou diviseur (Down Counter or Divider Config.vi), 27-4
- Configurer un compteur ou temps d'événement (Event or Time Counter Config.vi), 27-5
- Configurer un déclenchement numérique (Digital Trigger Config.vi), 25-7
- Configurer un générateur continu d'impulsions (Continuous Pulse Generator Config.vi), 27-3
- Configurer un générateur d'impulsions retardées (Delayed Pulse Generator Config.vi), 27-4
- Configurer un groupe numérique (Digital Group Config.vi), 25-4
- Configurer un mode numérique (Digital Mode Config.vi), 25-5
- Configurer une horloge numérique (Digital Clock Config.vi), 25-4
- Configurer une mesure de période ou une largeur d'impulsion (Pulse Width or Period Meas Config.vi), 27-8
- Constante booléenne, 5-6
- Constante chaîne de caractères, 6-24
- Constante de bibliothèque de VI, 11-31
- Constante de chemin, 11-31
- Constante de chemin du VI actuel, 11-30
- Constante de répertoire par défaut, 11-30
- Constante de répertoire temporaire, 11-31
- constantes arithmétiques définies par l'utilisateur, 4-9
- constantes supplémentaires définies par l'utilisateur, 4-22
- Construire un chemin, 11-7
- Construire un tableau, 7-4
- Construire un tableau de clusters, 8-4
- Contrôle de l'aide en ligne, 12-7
- Contrôle de la fenêtre d'aide, 12-7
- Contrôle RTSI (RTSI Control.vi), 29-18
- Contrôler un buffer numérique (Digital Buffer Control.vi), 25-3
- Conversion des unités de spectre (Spectrum Unit Conversion.vi), 40-7
- Convertir un buffer de thermocouple (Convert Thermocouple Buffer.vi), 30-9
- Convertir une mesure de jauge de contrainte (Convert Strain Gauge Reading.vi), 30-3
- Convertir une mesure de RTD (Convert RTD Reading.vi), 30-2
- Convertir une mesure de thermistance (Convert Thermistor Reading.vi), 30-7
- Convertir une mesure de thermocouple (Convert Thermocouple Reading.vi), 30-9
- Convolution, 39-5, 41-4
- Copier, 11-17
- Corrélation croisée (CrossCorrelation.vi), 39-7
- Cosécante, 4-15
- Cosinus, 4-15
- Cosinus inverse, 4-16
- Cosinus inverse hyperbolique, 4-16
- Cotangente, 4-15
- Coupe-bande équi-ondulation (Equipripple BandStop.vi), 41-6
- Créer des paramètres AppleEvent avec des spécificateurs d'objets, 52-15
- Créer un notificateur (Create Notifier.vi), 13-10
- Créer un rendez-vous (Create Rendezvous.vi), 13-16
- Créer un sémaphore (Create Semaphore.vi), 13-19
- Créer une file d'attente (Create Queue.vi), 13-13
- Créer une matrice complexe spéciale (Create Special Complex Matrix.vi), 45-10

Créer une matrice spéciale (Create Special Matrix.vi), 45-10
 CTR-Configurer un buffer (CTR Buffer Config.vi), 28-2
 CTR-Configurer un groupe (CTR Group Config.vi), 28-3
 CTR-Configurer un mode (CTR Mode Config.vi), 28-4
 CTR-Configurer une impulsion (CTR Pulse Config.vi), 28-10
 CTR-Contrôler (CTR Control.vi), 28-11
 CTR-Lire un buffer (CTR Buffer Read.vi), 28-3

D

Dans la gamme ?, 9-8
 Date/heure en secondes, 10-7
 DDE Advise Check, 50-2
 DDE Advise Start, 50-2
 DDE Advise Stop, 50-2
 DDE Close Conversation, 50-2
 DDE Execute, 50-3
 DDE Open Conversation, 50-3
 DDE Poke, 50-3
 DDE Request, 50-3
 DDE Srv Check Item, 50-4
 DDE Srv Register Item, 50-4
 DDE Srv Register Service, 50-4
 DDE Srv Set Item, 50-4
 DDE Srv Unregister Item, 50-5
 DDE Srv Unregister Service, 50-5
 Décalage logique, 13-5
 Décimer (Decimate.vi), 39-9
 Décimer un tableau 1D, 7-5
 Déconvolution (Deconvolution.vi), 39-9
 Définir les infos d'un périphérique DAQ (Set DAQ Device Information.vi), 29-19
 Définir les infos de l'élément de menu, 12-11
 Définir une occurrence, 13-22
 Démarrer un compteur (Counter Start.vi), 27-3
 Déplacer, 11-20
 Dérivée $x(t)$ (Derivative $x(t)$.vi), 39-10
 Désassembler, 8-6
 Désassembler par nom, 8-6
 Détection de crêtes (Peak Detector.vi), 40-5, 47-3
 Détection de crêtes avec seuillage (Threshold Peak Detector.vi), 40-7, 47-4
 Déterminant (Determinant.vi), 45-10
 Déterminant complexe (Complex Determinant.vi), 45-4
 Détruire un notificateur (Destroy Notifier.vi), 13-11
 Détruire un rendez-vous (Destroy Rendezvous.vi), 13-17
 Détruire un sémaphore (Destroy Semaphore.vi), 13-20
 Détruire une file d'attente (Destroy Queue.vi), 13-14
 DevClear, 35-2
 DevClearList, 35-5
 DIO-Attendre (DIO Wait.vi), 24-4
 DIO-Configurer (DIO Config.vi), 24-3
 DIO-Configurer un port (DIO Port Config.vi), 25-2
 DIO-Démarrer (DIO Start.vi), 24-4
 DIO-Ecrire (DIO Write.vi), 24-5
 DIO-Ecrire sur un port (DIO Port Write.vi), 25-2
 DIO-Lire (DIO Read.vi), 24-3
 DIO-Lire un port (DIO Port Read.vi), 25-2
 DIO-Lire/écrire une fois (DIO Single Read/Write.vi), 24-4
 DIO-Paramètre (DIO Parameter.vi), 25-6
 DIO-Réinitialiser (DIO Clear.vi), 24-3
 Diriger des signaux (Route Signal.vi), 29-18
 Distribution de type F (F Distribution.vi), 44-6
 Distribution de type T inv. (Inv T Distribution.vi), 44-9
 Distribution en χ^2 (Chi Square Distribution.vi), 44-5

Distribution en chi2 inv (Inv Chi Square Distribution.vi), 44-8
 Distribution normale inv. (Inv Normal Distribution.vi), 44-9
 dma – Etablir le mode DMA ou le mode d’E/S programmées, 34-10
 Droits d’accès, 11-16
 DSA-Etalonner (DSA Calibrate.vi), 29-11
 DSP2200-Configurer (DSP2200 Configure.vi) (Windows), 29-12
 DSP2200-Etalonner (DSP2200 Calibrate.vi) (Windows), 29-12

E

Echelle 1D (Scale 1D.vi), 46-6
 Echelle 2D (Scale 2D.vi), 46-6
 Echelle rapide 1D (Quick Scale 1D.vi), 46-5
 Echelle rapide 2D (Quick Scale 2D.vi), 46-6
 Eclater un nombre, 13-6
 Eclater un tableau 1D, 7-8
 Eclater une chaîne de caractères, 6-16
 Ecrire dans un fichier I16 (Write to I16 File.vi), 11-15
 Ecrire dans un fichier SGL (Write to SGL File.vi), 11-16
 Ecrire un buffer numérique (Digital Buffer Write.vi), 25-3
 Ecrire une clé (booléen) (Write Key (Boolean).vi), 11-27
 Ecrire une clé (chaîne de caractères) (Write Key (String).vi), 11-29
 Ecrire une clé (chemin) (Write Key (Path).vi), 11-28
 Ecrire une clé (double) (Write Key (Double).vi), 11-28
 Ecrire une clé (I32) (Write Key (I32).vi), 11-28
 Ecrire une clé (U32) (Write Key (U32).vi), 11-29
 Ecrire une seule valeur numérique (Digital Single Write.vi), 25-7
 Effacer, 11-17
 Egal à 0 ?, 9-7
 Egaux ?, 9-7
 Élément de menu sélectionné, 12-10
 En complexe double précision, 4-13
 En complexe précision étendue, 4-13
 En complexe simple précision, 4-13
 En données G, 51-4
 En entier de type mot, 4-14
 En entier de type mot non signé, 4-14
 En entier de type octet, 4-12
 En entier long, 4-13
 En entier long non signé, 4-14
 En entier octets non signé, 4-14
 En flottant double précision, 4-13
 En flottant précision étendue, 4-13
 En flottant simple précision, 4-14
 En majuscules, 6-17
 En minuscules, 6-17
 EnableLocal, 35-5
 EnableRemote, 35-5
 Envoyer une notification (Send Notification.vi), 13-11
 EOF, 11-18
 erf(x), 44-6
 erfc(x), 44-6
 Erreur quadratique moyenne (MSE.vi), 44-11
 Espace blanc ?, 9-11
 Estimation AC & DC (AC & DC Estimator.vi), 40-2
 Estimation de puissance et fréq. (Power & Frequency Estimate.vi), 40-5
 ET, 5-3
 ET logique entre les éléments du tableau, 5-3
 Etat de la fenêtre d’aide, 12-8
 Evaluation linéaire 1D (1D Linear Evaluation.vi), 46-2
 Evaluation linéaire 2D (2D Linear Evaluation.vi), 46-3
 Evaluation polynomiale 1D (1D Polynomial Evaluation.vi), 46-2

Evaluation polynomiale 2D (2D Polynomial Evaluation.vi), 46-3
 Exponentiel, 4-19
 Exponentiel (Arg)-1, 4-19

F

Factorisation de Cholesky (Cholesky Factorization.vi), 45-3
 Factorisation de Cholesky complexe (Complex Cholesky Factorization.vi), 45-4
 Factorisation LU (LU Factorization.vi), 45-12
 Factorisation LU complexe (Complex LU Factorization.vi), 45-7
 Factorisation QR (QR Factorization.vi), 45-14
 Factorisation QR complexe (Complex QR Factorization.vi), 45-9
 Factorisation SVD (SVD Factorization.vi), 45-17
 Factorisation SVD complexe (Complex SVD Factorization.vi), 45-9
 Fenêtre à profil plat (Flat Top Window.vi), 42-4
 Fenêtre cosinus générale (General Cosine Window.vi), 42-4
 Fenêtre de Blackman (Blackman Window.vi), 42-1
 Fenêtre de Blackman exacte (Exact Blackman Window.vi), 42-3
 Fenêtre de Blackman-Harris (Blackman-Harris Window.vi), 42-2
 Fenêtre de force (Force Window.vi), 42-4
 Fenêtre de Hamming (Hamming Window.vi), 42-5
 Fenêtre de Hanning (Hanning Window.vi), 42-5
 Fenêtre de Kaiser-Bessel (Kaiser-Bessel Window.vi), 42-6
 Fenêtre en pic de cosinus (Cosine Tapered Window.vi), 42-2
 Fenêtre exponentielle (Exponential Window.vi), 42-3
 Fenêtre temporelle mise à l'échelle (Scaled Time Domain Window.vi), 40-7
 Fenêtre triangulaire (Triangle Window.vi), 42-6
 Fermer l'application ou la référence de VI, 12-3
 Fermer les données de Config (Close Config Data.vi), 11-25
 Fermer refnum Automation, 51-3
 Fermer un fichier, 11-7
 FFT complexe (Complex FFT.vi), 39-3
 FFT complexe inverse (Inverse Complex FFT.vi), 39-13
 FFT inverse réelle (Inverse Real FFT.vi), 39-16
 FFT réelle (Real FFT.vi), 39-18
 FHT, 39-12
 FHT inverse (Inverse FHT.vi), 39-15
 Filtre bande étroite RIF (FIR Narrowband Coefficients.vi), 41-9
 Filtre de Bessel (Bessel Filter.vi), 41-2
 Filtre de Butterworth (Butterworth Filter.vi), 41-3
 Filtre de Chebyshev (Chebyshev Filter.vi), 41-4
 Filtre de Chebyshev inv. (Inverse Chebyshev Filter.vi), 41-13
 Filtre elliptique (Elliptic Filter.vi), 41-5
 Filtre fenêtré RIF (FIR Windowed Filter.vi), 41-10
 Filtre médian (Median Filter.vi), 41-13
 Filtre RII (IIR Filter.vi), 41-11
 Filtre RII avec circuit intégré (IIR Filter with Integrated Circuit.vi), 41-12
 Filtre RII cascadié (IIR Cascade Filter.vi), 41-10
 Filtre RII cascadié avec circuit intégré (IIR Cascade Filter with Integrated Circuit.vi), 41-11
 Filtres bande étroite RIF (FIR Narrowband Filter.vi), 41-9
 Fin de ligne, 6-24

FindLstn, 35-7
FindRQS, 35-5
Fonction "Appeler une fonction d'une DLL",
13-3
Fonction de réseau (moy) (Network Functions
(avg).vi), 40-4
Fonction de transfert (Transfer Function.vi),
40-8
Fonction réponse impulsionnelle (Impulse
Response Function.vi), 40-4
Format de la chaîne de caractères de la
date/heure, 10-7
Formater & Dépouiller, 6-19
Formater en une chaîne de caractères, 6-8

G

Générer un train d'impulsions (Generate Pulse
Train.vi), 26-3
Générer une impulsion retardée (Generate
Delayed Pulse.vi), 26-2
Générer une occurrence, 13-21
Gestionnaire général d'erreurs, 10-12
Gestionnaire simple d'erreurs, 10-13
Get Target ID, 52-3, 52-4, 53-3
GPIB Clear, 34-3
GPIB Initialization, 34-4
GPIB Misc, 34-4
GPIB Read, 34-5
GPIB Serial Poll, 34-6
GPIB Status, 34-6
GPIB Trigger, 34-6
GPIB Wait, 34-6
GPIB Write, 34-7
gts – Passer du contrôleur actif à l'état de
veille, 34-11

H

Histogramme, 44-7

Histogramme général (General Histogram.vi),
44-6

I

ICTR-Contrôler (ICTRControl.vi), 28-11
Implique, 5-4
Imprimable ?, 9-10
Imprimer la face-avant (Print Panel.vi), 12-5
Indexer & Ajouter, 6-9
Indexer & Assembler un tableau de clusters,
8-5
Indexer & Dépouiller, 6-10
Indexer un tableau, 7-5
Inférieur ?, 9-8
Inférieur à 0 ?, 9-9
Inférieur ou égal ?, 9-8
Inférieur ou égal à 0 ?, 9-8
Infos du volume, 11-23
Infos sur le fichier/répertoire, 11-18
Infos sur un raccourci de menus, 12-10
Initialiser un tableau, 7-6
Insérer des éléments de menus, 12-10
Insérer un élément de file d'attente (Insert
Queue Element.vi), 13-15
Intégrale $x(t)$ (Integral $x(t)$.vi), 39-13
Intégration numérique (Numeric
Integration.vi), 47-2
Interpolant spline (Spline Interpolant.vi), 43-6
Interpolation polynomiale (Polynomial
Interpolation.vi), 43-5
Interpolation rationnelle (Rational
Interpolation.vi), 43-5
Interpolation spline (Spline Interpolation.vi),
43-8
Intervertir les mots, 13-7
Intervertir les octets, 13-7
Inverse sinus, 4-17
IP en chaîne de caractères, 48-3
ist – Définir le bit d'état individuel, 34-12

L

Libérer un sémaphore (Release Semaphore.vi), 13-20

Lire dans un fichier I16 (Read from I16 File.vi), 11-15

Lire dans un fichier SGL (Read from SGL File.vi), 11-15

Lire des caractères dans un fichier (Read characters from File.vi), 11-8

Lire un buffer numérique (Digital Buffer Read.vi), 25-3

Lire un compteur (Counter Read.vi), 27-3

Lire un fichier, 11-8

Lire un port numérique (Read from Digital Port.vi), 23-2

Lire une clé (booléen) (Read Key (Boolean).vi), 11-25

Lire une clé (chaîne de caractères) (Read Key (String).vi), 11-26

Lire une clé (chemin) (Read Key (Path).vi), 11-26

Lire une clé (double) (Read Key (Double).vi), 11-25

Lire une clé (I32) (Read Key (I32).vi), 11-26

Lire une clé (U32) (Read Key (U32).vi), 11-26

Lire une ligne numérique (Read from Digital Line.vi), 23-2

Lire une seule valeur numérique (Digital Single Read.vi), 25-7

Lissage de phase (Unwrap Phase.vi), 39-19

Lister un répertoire, 11-19

llo – Verrouillage local, 34-12

loc – Atteindre local, 34-8

loc – Mettre le contrôleur en état local, 34-12

Logarithme de base 10, 4-19

Logarithme de base 2, 4-19

Logarithme de base X, 4-19

Logarithme naturel, 4-20

Logarithme naturel (Arg +1), 4-20

Longueur d'une chaîne de caractères, 6-17

LPM-16-Etalonner (LPM-16 Calibrate.vi), 29-15

M

Make Alias, 52-14

MakeAddr, 35-12

Mantisse & Exposant, 13-5

Matrice complexe définie positive-test (Test Complex Positive Definite.vi), 45-17

Matrice définie positive-test (Test Positive Definite.vi), 45-18

Matrice inverse (Inverse Matrix.vi), 45-11

Matrice inverse complexe (Complex Inverse Matrix.vi), 45-6

Matrice pseudo-inverse complexe (Complex PseudoInverse Matrix.vi), 45-9

Max & Min, 9-9

Max & Min d'un tableau, 7-3

Médiane (Median.vi), 44-10

Mesurer une fréquence (Measure Frequency.vi), 26-3

Mesurer une largeur d'impulsion ou une période (Measure Pulse Width or Period.vi), 26-4

Mettre à jour le fichier, 11-19

MIO-Configurer (MIO Config.vi) (Windows), 29-17

MIO-Etalonner (MIO Calibrate.vi) (Windows), 29-16

Modes de comparaison, 9-3

Moment centré (Moment About Mean.vi), 44-10

Motif d'impulsion (Pulse Pattern.vi), 38-5

Motif d'impulsion Dirac (Impulse Pattern.vi), 38-4

Motif de modulation de fréquence (Chirp Pattern.vi), 38-3

Motif de rampe (Ramp Pattern.vi), 38-5

Motif sinc (Sinc Pattern.vi), 38-7

Motif sine (Sine Pattern.vi), 38-7

Moyenne (Mean.vi), 44-9

N

Nœud d'appel par référence, 12-2

Nœud de méthode, 12-3, 51-3

Nœud de propriété (Fonctions d'Automation ActiveX), 51-3

Nœud de propriété (Fonctions de contrôle d'applications), 12-5

Nombre conditionnel d'une matrice (Matrix Condition Number.vi), 45-13

Nombre conditionnel d'une matrice complexe (Complex Matrix Condition Number.vi), 45-7

Nombre en tableau de booléens, 5-5

Non, 5-4

Non égal à 0 ?, 9-10

Non égaux ?, 9-10

Non ET, 5-5

Non OU, 5-5

Non OU exclusif, 5-5

Normaliser un vecteur (Normalize Vector.vi), 46-4

Normaliser une matrice (Normalize Matrix.vi), 46-4

Norme d'une matrice (Matrix Norm.vi), 45-13

Norme d'une matrice complexe (Complex Matrix Norm.vi), 45-7

Nouveau fichier, 11-20

Nouveau répertoire, 11-20

O

Obtenir des informations sur le matériel SCXI (Get SCXI Information.vi), 29-15

Obtenir des informations sur un périphérique DAQ (Get DAQ Device Information.vi), 29-15

Obtenir des infos d'échelle (Get Scale Information.vi), 29-21

Obtenir des infos de voies (Get Channel Information.vi), 29-20

Obtenir des infos sur un élément de menu, 12-9

Obtenir l'état d'un notificateur (Get Notifier Status.vi), 13-11

Obtenir l'état d'un rendez-vous (Get Rendezvous Status.vi), 13-17

Obtenir l'état d'un sémaphore (Get Semaphore Status.vi), 13-20

Obtenir l'état d'une file d'attente (Get Queue Status.vi), 13-14

Obtenir la chaîne de caractères de la date/heure, 10-9

Obtenir la date et l'heure en secondes, 10-9

Obtenir les noms de voies DAQ (Get DAQ Channel Names.vi), 29-20

off – Mettre l'appareil hors ligne, 34-8

off – Mettre le contrôleur hors ligne, 34-13

Opérateur arithmétique, 5-4

OU, 5-5

OU exclusif, 5-4

OU logique entre les éléments du tableau, 5-6

Ouvrir les données de Config (Open Config Data.vi), 11-25

Ouvrir refnum Automation, 51-2

Ouvrir un fichier, 11-20

Ouvrir une référence d'application, 12-3

Ouvrir une référence de VI, 12-4

Ouvrir/Créer/Remplacer un fichier (Open/Create/Replace File.vi), 11-8

P

Paramètres d'impulsion (Pulse Parameters.vi), 40-6

Parks-McClellan, 41-14

Pas un chemin, 11-30

Pas un nombre/chemin/refnum ?, 9-10

Pas un notificateur (Not a Notifier.vi), 13-11

Pas un refnum, 11-30

- Pas un rendez-vous (Not a Rendezvous.vi), 13-17
- Pas un sémaphore (Not a Semaphore.vi), 13-20
- Pas une file d'attente (Not a Queue.vi), 13-15
- PassControl, 35-3
- Passe-bande équi-ondulation (Equiripple BandPass.vi), 41-6
- Passe-bas équi-ondulation (Equiripple LowPass.vi), 41-7
- Passe-haut équi-ondulation (Equiripple HighPass.vi), 41-6
- pct – Transmettre le contrôle, 34-8
- Piocher une ligne & Ajouter, 6-13
- Polaire vers cartésien 1D (1D Polar To Rectangular.vi), 46-2
- Port d'entrée (In Port.vi) (Windows 3.1 et Windows 95), 13-8
- Port de sortie (Out Port.vi) (Windows 3.1 et Windows 95), 13-8
- ppc – Configurer l'interrogation parallèle (activer et désactiver), 34-13
- ppc – Configurer l'interrogation parallèle (parallel poll), 34-9
- PPC Accept Session, 53-2
- PPC Browser, 52-5
- PPC Close Port, 53-3
- PPC End Session, 53-3
- PPC Inform Session, 53-3
- PPC Open Port, 53-3
- PPC Read, 53-4
- PPC Start Session, 53-4
- PPC Write, 53-5
- PPoll, 35-6
- PPollConfig, 35-3
- PPollUnconfig, 35-6
- ppu – Déconfigurer l'interrogation parallèle (parallel poll), 34-13
- PREFIXE Arborescence de VI (PREFIX VI Tree.vi), 32-7
- PREFIXE Auto-test (PREFIX Self-Test.vi), 32-6
- PREFIXE Fermer (PREFIX Close.vi), 32-2
- PREFIXE Initialiser (PREFIX Initialize.vi) et PREFIXE Initialiser (VXI, basé registre) (PREFIX Initialize (VXI, Reg-based).vi), 32-3
- PREFIXE Initialiser le nettoyage de l'utilitaire (PREFIX Utility Clean UP Initialize.vi), 32-6
- PREFIXE Installer l'instrument par défaut de l'utilitaire (PREFIX Utility Default Instrument Setup.vi), 32-7
- PREFIXE Message d'erreur (PREFIX Error Message.vi), 32-3
- PREFIXE Modèle basé message (PREFIX Message-Based Template.vi), 32-5
- PREFIXE Modèle basé registre (PREFIX Register-Based Template.vi), 32-5
- PREFIXE Recherche d'erreur (PREFIX Error query.vi), Recherche d'erreurs (multiples.vi) (Error Query (Multiple).vi) et Message d'erreur (Error Message.vi), 32-3
- PREFIXE Recherche de révision (PREFIX Revision Query.vi), 32-6
- PREFIXE Réinitialiser (PREFIX Reset.vi), 32-5
- Produit externe (Outer Product.vi), 45-13
- Produit externe complexe (Complex Outer Product.vi), 45-8
- Produit scalaire (Dot Product.vi), 45-11
- Produit scalaire complexe (Complex Dot Product.vi), 45-5
- Puissance croisée (Cross Power.vi), 39-6
- Puissance de 10, 4-20
- Puissance de 2, 4-20
- Puissance de X, 4-20

Q

Quitter LabVIEW, 12-6

R

Racines polynomiales complexes (Complex Polynomial Roots.vi), 47-2
 Rang d'une matrice (Matrix Rank.vi), 45-13
 Rang d'une matrice complexe (Complex Matrix Rank.vi), 45-8
 RcvRespMsg, 35-10
 ReadStatus, 35-3
 Receive, 35-3
 ReceiveSetup, 35-10
 Rechercher, 11-22
 Rechercher dans un tableau 1D, 7-7
 Rechercher un symbole, 6-15
 Rechercher une expression, 6-10
 Redimensionner un rendez-vous (Resize Rendezvous.vi), 13-17
 Redimensionner un tableau, 7-7
 Redresser à partir d'une chaîne de caractères, 13-8
 Refnum en chemin, 11-21
 Regrouper des nombres, 13-4
 Réinitialiser un périphérique (Device Reset.vi), 29-11
 Remplacer un élément d'un tableau, 7-6
 Renverser un tableau 1D, 7-7
 Renverser une chaîne de caractères, 6-13
 ResetSys, 35-7
 Résolution d'équations linéaires (Solve Linear Equations.vi), 45-16
 Résolution d'équations linéaires complexes (Solve Complex Linear Equations.vi), 45-15
 Retour à la ligne, 6-24
 Retour chariot, 6-24
 Rotation, 13-5
 Rotation dans un tableau 1D, 7-7
 Rotation dans une chaîne de caractères, 6-13
 Rotation droite avec retenue, 13-6
 Rotation gauche avec retenue, 13-5
 rpp – Effectuer une interrogation parallèle (parallel poll), 34-13

rsc – Demander ou affranchir le contrôle du système, 34-14
 rsv – Demander un service et/ou activer l'octet d'état d'interrogation série (serial poll), 34-14

S

SCXI-Balayage de température (SCXI Temperature Scan.vi), 30-11
 Secondes en date/heure, 10-10
 Sélectionner, 9-11
 Sélectionner & Ajouter, 6-16
 Sélectionner & Dépouiller, 6-16
 Send, 35-4
 SendCmds, 35-11
 SendDataBytes, 35-11
 SendIFC, 35-8
 SendList, 35-6
 SendLLO, 35-8
 SendSetup, 35-11
 Serial Port Break, 36-2
 Serial Port Init, 36-2
 Serial Port Read, 36-2
 Serial Port Write, 36-2
 SetRWLS, 35-8
 SetTimeout, 35-12
 Seuil d'un tableau 1D, 7-8
 sic – Envoyer une réinitialisation d'interface, 34-14
 Signal arbitraire (Arbitrary Wave.vi), 38-2
 Signal carré (Square Wave.vi), 38-8
 Signal en dents de scie (Sawtooth Wave.vi), 38-6
 Signal sinusoïdal (Sine Wave.vi), 38-8
 Signal triangulaire (Triangle Wave.vi), 38-9
 Sinc, 4-18
 Sinus, 4-18
 Sinus hyperbolique, 4-16
 Sous-ensemble d'un tableau, 7-4

Sous-ensemble d'une chaîne de caractères, 6-17
 Spectre d'amplitude et de phase (Amplitude et Phase Spectrum.vi), 40-2
 Spectre de puissance (Power Spectrum.vi), 39-17
 Spectre de puissance auto (Auto Power Spectrum.vi), 40-2
 Spectre de puissance croisé (Auto Power Spectrum.vi), 40-3
 sre – Etablir ou réinitialiser l'activation à distance, 34-15
 structure Condition, 3-2
 structure Séquence, 3-2
 Supérieur ?, 9-7
 Supérieur à 0 ?, 9-7
 Supérieur ou égal ?, 9-7
 Supérieur ou égal à 0 ?, 9-7
 Supprimer des éléments de menus, 12-8
 Supprimer un élément de file d'attente (Remove Queue Element.vi), 13-15
 Supprimer une clé (Remove Key.vi), 11-27
 Supprimer une section (Remove Section.vi), 11-27

T

Tab, 6-24
 Table de contingence (Contingency Table.vi), 44-5
 Tableau d'octets en une chaîne de caractères, 6-22
 Tableau de booléens en nombre, 5-3
 Tableau de chaînes de caractères en chemin, 11-17
 Tableau de chaînes en chemin, 6-22
 Tableau en cluster, 7-4, 8-4
 Tableau en une chaîne de caractères au format tableur, 6-7
 Taille d'un tableau, 7-4
 Tangente hyperbolique, 4-16

Tangente inverse, 4-17
 Tangente inverse (deux entrées), 4-17
 TCP Close Connection, 48-3
 TCP Create Listener, 48-3
 TCP Listen, 48-1
 TCP Open Connection, 48-3
 TCP Read, 48-4
 TCP Wait on Listener, 48-4
 TCP Write, 48-4
 TestSRQ, 35-9
 TestSys, 35-9
 Trace, 45-18
 Trace d'une matrice complexe (Complex Matrix Trace.vi), 45-8
 Transformée de Hilbert rapide (Fast Hilbert Transform.vi), 39-11
 Transformée de Hilbert rapide inverse (Inverse Fast Hilbert Transform.vi), 39-14
 Transposer un tableau 2D, 7-9
 Trigger, 35-4
 TriggerList, 35-6
 Trouver la première erreur (Find First Error.vi), 10-11
 Type de chemin, 11-21
 Type et Créateur, 11-22

U

UDP Close, 49-1
 UDP Open, 49-1
 UDP Read, 49-2
 UDP Write, 49-2

V

Valeurs et vecteurs propres (EigenValues & Vectors.vi), 45-11
 variable globale, 3-3
 variable locale, 3-4
 Variance (Variance.vi), 44-13
 Vecteur unité (Unit Vector.vi), 46-7

Vecteurs et valeurs propres complexes
(Complex EigenValues & Vectors.vi), 45-5

Verrouiller la gamme, 11-19

VI Corrélation croisée (CrossCorrelation.vi),
39-7

VI $Y[i] = X[i-n]$, 39-19

Vider une file d'attente (Flush Queue.vi),
13-14

VISA Assert Trigger, 33-5

VISA Attribute Node, 33-19

VISA Clear, 33-5

VISA Close, 33-5

VISA Disable Event, 33-10

VISA Discard Events, 33-11

VISA Enable Event, 33-11

VISA Find Resource, 33-6

VISA Find Resources simple, 33-4

VISA Flush Serial Buffer, 33-18

VISA In8 / In16 / In32, 33-12

VISA Lock, 33-6

VISA Map Address, 33-16, 33-18

VISA Memory Allocation, 33-13, 33-17

VISA Memory Free, 33-13, 33-17

VISA Move In8 / Move In16 / Move In32,
33-14

VISA Move Out8 / Move Out16 / Move
Out32, 33-14

VISA Open, 33-7

VISA Out8 / Out16 / Out32, 33-15

VISA Peek8 / Peek16 / Peek32, 33-17

VISA Poke8 / Poke16 / Poke32, 33-17

VISA Read, 33-8

VISA Read simple, 33-4

VISA Read STB, 33-9

VISA Serial Write and Read simple, 33-4

VISA Set Serial Buffer Size, 33-19

VISA Status Description, 33-9

VISA Unlock, 33-9

VISA Unmap Address, 33-18

VISA Wait On Event, 33-11

VISA Write, 33-9

VISA Write and Read simple, 33-4

VISA Write simple, 33-5

Voies en indice (Channel to Index.vi), 29-7

W

Wait for RQS du GPIB, 34-7

WaitSRQ, 35-9

Y

$Y[i] = \text{Clip} \{X[i]\}$, 39-19

$Y[i] = X[i-n]$, 39-19