RACAL
The Electronics Group

User Manual

Model 6062
Precision Analog
Input/Output Module

P/N: 33-1070-99999
Version : 2.0 without drawing
April 2002

RACAL

TABLE OF CONTENTS

GETTING STARTED

For your safety

This equipment may retain DC residual charges, which can cause serious injury.
If the module is used in a chassis connected to the AC network, make sure that all metallic parts accessible to the user are properly grounded.
The use of the chassis/module set with an ungrounded power cord is prohibited.

If the instrument shows any signs of electrical shock or is stored in unfavorable conditions, it should not be used until qualified personnel have inspected it.

Warning :
Before switching on your chassis, YOU MUST SYSTEMATICALLY check that user connectors are plugged in.

Units operate from high internal voltages. DISCONNECT FROM THE POWER source before removing conver.

Product warranty

All products sold by RACAL SYSTEMS Electronique SA are guaranteed under the following terms:

With the reservations that will follow, RACAL SYSTEMS Electronique SA undertakes to correct, either by repair or replacement, any material defect, which appears during the year following the delivery of the instrument. The device will first undergo an inspection to determine the cause of the dysfunction, and the validity of the operating environment.
The reservations mentioned above are the following:
a) Some components and accessories are not by nature intended to operate during an entire year. If one or more of those components or accessories manufactured by RACAL SYSTEMS Electronique S.A. was defective upon delivery, or if the defect appeared within a reasonable time for a reasonable use, RACAL SYSTEMS Electronique S.A. agrees to repair or replace said components or accessories once RACAL SYSTEMS Electronique S.A. has obtained all the factors inherent to the conditions of use, and after said components or accessories have been returned to RACAL SYSTEMS Electronique S.A post paid.

b) All components declared defective must be returned to RACAL SYSTEMS Electronique S.A. post paid, and shall be returned free of charge to the customer if RACAL SYSTEMS Electronique S.A. has in fact the defect mentioned by the customer.

c) The RACAL SYSTEMS Electronique S.A company does not guarantee all components or accessories, which it did not manufacture. Nonetheless, if one of said components or accessories should prove defective, RACAL SYSTEMS Electronique S.A. agrees to assist the client in obtaining either replacement or repair of said components or accessories from the manufacturer in question, on the condition that the defects be covered by the manufacturer's own warranty.

d) RACAL SYSTEMS Electronique S.A. shall not be held responsible for any repairs or modifications undertaken by anyone other than those authorized by RACAL SYSTEMS Electronique S.A.

RACAL SYSTEMS Electronique S.A. disclaims all responsibilities to its clients, dealers or distributors concerning its products, for damages of any form whatsoever, which arise from areas other than the manufacture, sale, transportation, repair, maintenance or replacement, or any other cause brought about by an abnormal use of the product.

Proprietary Information

This document as well as the technical data attached to it are the property of RACAL SYSTEMS Electronique S.A. and may not, without the express written consent of RACAL SYSTEMS Electronique S.A., be used in whole or in part by any competing company, or be used for production by any company other than RACAL SYSTEMS Electronique S.A. The information contained in this document is derived from research of which RACAL SYSTEMS Electronique S.A. is the sole proprietor and should not serve any purpose except for the use of the device, maintenance operations and technical evaluations necessary to incorporate this product into a specific system.

Unpacking and Inspection

Before unpacking the module, verify that no traces of damage are visible on the packing box. Any irregularities must be mentioned on the bill of lading. Remove the module from of its box, and keep the box in case it should be needed later. Immediately notify the carrier in case of any problems. Have the instrument tested by qualified personnel before applying power.

Instructions for return under warranty

Use the original packaging when you return the instrument to RACAL Systems S.A. for warranty return or adjustments. The box and the module's anti-static bag provide secure packaging. If the original packaging is not re-useable, use an anti-static plastic package, as well as bubble plastic to protect the module during transport.

1. Initial Contact

Overview

This document provides the information necessary to install and correctly operate one or many 6062 modules inside a 9; ,bus-compatible chassis.

The 6062 is an un-isolated, fast, high-precision data acquisition and generation module. All the input and output lines are referenced to the system ground.

The 6062 is a C-size module used in industrial process control and measure applications requiring:

- a compact unit providing A/D and D/A converters: eight independent analog channels
- differential analogue inputs: signal measuring with significant noise in common modes
- high accuracy and resolution: 16-bits
- analog output generators in 4-wire mode
- large sample storage capacity for generation and acquisition: 64k samples per channel
- a large range of sampling frequencies: from 1 to 100,000 samples per second
- two communication modes: Message-based and Register-based
- support for internal, external, daisy-chain, synchronous and asynchronous triggers
- coupling of many channels to increase the sampling size and speed: 1 channel of 400,000 samples/second and up to 256k samples deep
- continuous operating modes on one or many channels
- modification of certain samples without desynchronizing sample generation

Features (version 6062A)

16-bit resolution for analog inputs/outputs
Four voltage outputs in 4-wire mode
4 high-impedance differential inputs
64k-deep samples per channel
Simple C-size module
Entirely compatible with revision 1.4 the VXI bus standard
SCPI compatible
,(((▯▯▯▯▯compatible

16-bit resolution on analog input/output channels

Modules covered in this manual

Module 6062A provides four analog outputs and four analog inputs, with 16-bit accuracy.

Module 6062B-4 provides four analog outputs.

Module 6062B-8 provides eight analog outputs.

Module 6062C-4 provides four analog inputs.

Module 6062C-8 provides eight analog inputs.

Note The 6062 series are not isolated modules, this implies the channels are referenced to the system ground.

Module Ordering Information

ID number	Model	Option	Description
33-1070-99999	6062	ALL	User manual
33-1070-00000	6062A	Standard	Four Analog Outputs ±10V Four Analog Inputs ±10V
33-1071-00000	6062B4	Standard	Four Analog Outputs ±10V
33-1072-00000	6062B8	Standard	Eight Analog Outputs ±10V
33-1073-00000	6062C4	Standard	Four Analog Inputs ±10V
33-1074-00000	6062C8	Standard	Eight Analog Inputs ±10V
33-1075-00000	6062B4 Opt 06	Standard	Four Outputs ±25V, Isolated
33-1075-C0000	6062B4 Opt 06	Option S2	Four Outputs ±40V, Isolated 2 by 2
33-107X-10000	6062X	Option 01	512k Words Mass Memory
33-107X-20000	6062X	Option 02	One or Two Inputs ±10V, 400KS/s
33-107X-30000	6062X	Option 03	Four or Eight Outputs ±10V, 2.5MS/s
33-107X-40000	6062X	Option 04	Four or Eight Anti-Aliasing Filters
33-107X-50000	6062X	Option 05	Four or Eight Inputs ±10V, 250KS/s
33-107X-70000	6062X	Option 07	One or Two Inputs ±10V, 1MS/s
33-107X-A0000	6062X	Option 10	Edge Triggering for outputs channels only
33-107X-B0000	6062X	Option S3	Sampling at external clock frequency

General Characteristics

VXIbus Interface

Manufacturer ID:	4091 (0FFB hex.): RACAL-INSTRUMENT
Model 6062A :	1070
Model 6062B-4 :	1071
Model 6062B-8 :	1072
Model 6062C-4 :	1073
Model 6062C-8 :	1074
Logical Address :	Dynamic or Static (1 to 254)
Address Space :	A16/A32 : D16
Instrument Class :	Slave message-based instrument
Interrupt Levels :	Programmable I (0-7)
TTLTRIG Line:	Programmable

SUPPORTS WORD SERIAL PROTOCOL

Specifications Common to All Versions

Execution time in interactive mode:	< 5 ms per channel
Data Memory size:	64k samples per channel
Time-base period (standard):	from 10 µs to 1 second (with 10MHz internal clock)
Time-base period (Option 03):	from 400ns to 0.1 second (with 10MHz internal clock)

Triggers Capability

Trigger source:	By command, 1 to 8 TTLTRIG lines TTLTRIG (VXI bus) or External
Data memory sequences:	Start, Stop
Trigger modes:	IMMediate, COUNt
Trigger Action (standard version):	0 : Start, 1 : Stop
Trigger Action (Option 10):	Rising Edge : Stop then Start

External Clock inputs (J5)

Impedance:	>10kΩ
Overvoltage Protection:	100V continuous and RMS.
Frequency Range:	10Hz <= CLKIN <= 16MHz (square signal)
Duty Cycle:	50% ± 10%
Trigger Delay:	<50ns
Minimal Amplitude:	3V peak to peak (square signal)

Input triggers (J2, J7)

Number:	One per channel
Level:	TTL level referenced to the system ground
Impedance:	>50kΩ
Hysteresis:	100mV ±10%
Overvoltage Protection:	100V continuous
Min. time at low level	≥100ns

10V Reference Input for Self-Test (J3)

Impedance:	>100kΩ
Acceptable Levels:	10V ±0.05%
Overvoltage Protection:	100V continuous and RMS

10.000V Reference Output (J4)

Accuracy:	±0.1% (23°C ±2°C) (Iout = 1mA)
Thermal Drift:	50ppm / °C
Maximum Output Current:	20mA

Output triggers (J2, J7)

Number:	Two per channel (Programmable)
Fan Out	10 loads ALS TTL equivalent
Transition Time:	<100ns (Level 1 to 0 , capacitive load < 50pF)
Self-Test Coverage:	90 % to 25 °C
MTBF:	50,000 hours at 25 °C 40,000 hours at 35 °C
User Connectors:	POSITRONIC SGMC 20, 26 pins
Required Cooling:	4.0 l/s, 0.5 mm H_2O.
Operating Temperature:	10 °C to 50 °C.
Power Requirements	
+5 Vdc:	from 4.75 to 5.25 Vdc IPM ≤ 3 A, IDM ≤ 0.6 A
+24 Vdc:	from +22 Vdc to +26 Vdc IPM ≤ 0.5 A, IDM ≤ 0.4 A
with Option 06:	IPM ≤ 1.5 A, IDM ≤ 0.8 A
-24 Vdc:	from -22 Vdc to -26 Vdc IPM ≤ 0.5 A, IDM ≤ 0.4 A
with Option 06:	IPM ≤ 1.5 A, IDM ≤ 0.8 A
Weight:	approximately 2.5 kg

Analog output characteristics

Number of channels:	6062A : 4 6062B-4: 4 6062B-8: 8 6062C-4: 0 6062C-8: 0
Output Modes:	2-wire or 4-wire mode
Resolution:	16 bits
Monotonicity:	14 bits
Voltage Range:	±10V
Integral Non-linearity:	±0.006% FSR
Differential Non-linearity:	±0.006% FSR
Gain Accuracy and Offset:	±(0.01% FSR + 1mV) at 23°C ± 2°C
Thermal Drift:	±(0.002% FSR + 0.1mV)/°C max.
Settline Time (20V output step) :	10µs at 0.1%, 50µs at 0.01%
Small Signal Bandwidth:	>200kHz
Slew Rate:	>5V/µs
Sampling Rate:	1 to 100,000 S/s (Full spec.) Up to 500 kS/s (Limited spec.)
Noise Voltage (20 Hz to 20 MHz):	2mV RMS, 5mV pp
Output Resistance:	< 2Ω for 2-wire, <0.2Ω for 4-wire
Max. Output Current:	20mA per channel

Option 03 analog output characteristics :

Sampling Rate:	10 to 100,000 S/s (Full spec.) Up to 2.5 MS/s (Limited spec.)
Noise Voltage (20 Hz to 20 MHz):	5mV RMS, 20mV pp
Settling time (0.1%) :	6µs typ. for 20V output step 4µs typ. for 1LSB output step
Slew rate :	10V/µs typ.

Analog input characteristics

Number of channels:	6062A : 4 6062B-4: 0 6062B-8: 0 6062C-4: 4 6062C-8: 8
Input Types:	Differential or simple ended
Max. Voltage in Common Mode:	±10V
Resolution:	16 bits
No Missing Code:	15 bits
Voltage Range:	±10V
Integral Non-linearity:	±0.005% FSR
Differential Non-linearity:	±0.005% FSR
Gain Accuracy and Offset:	±(0.01% FSR+1mV) at 23°C ± 2°C
Thermal Drift:	±(0.002% FSR + 0.1mV)/°C max.
Digitalization Noise:	±(0.01% FSR ± 1mV)
Input Impedance:	approx. 10MΩ // 100pF
Overvoltage Protection:	250V continuous or RMS (50/60Hz)
Recovery Time : (after overvoltage)	<10ms
Set-up Time:	10µs at 0.1%, 50µs at 0.01%
Small Signal Bandwidth:	>200kHz
Sampling Rate:	1 to 100,000 S/s
Rejection in Common Mode:	80dB (DC) 70dB (AC 50/60 Hz)

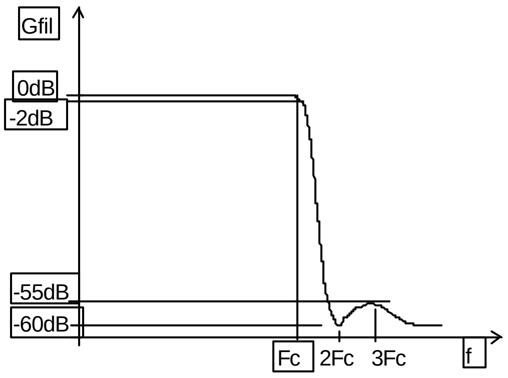
Option 05 analog input characteristics

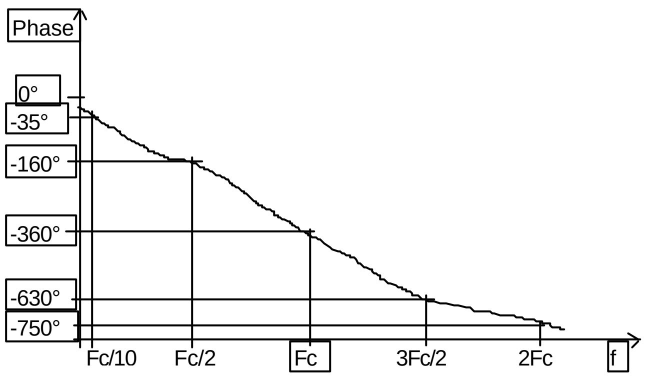
Sampling Rate:	10 to 250,000 S/s
Digitalization Noise:	±(0.05% FSR ± 5mV)

Anti-aliasing Filter Specification (Opt 04)

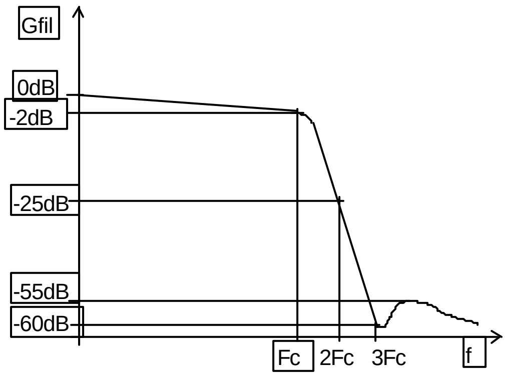
Number of filter :	4 independent filters
Filter Order :	8th Order with programmable Corner Frequency, followed by a 4th Order fixed Corner Frequency filter (200kHz)
Type (programmable) :	- Elliptic - Pseudo-linear Phase - Bypass (4th Order Filter only)
Corner Frequency (Fc) :	5Hz to 50kHz
Modes :	- Fc locked to Sampling Frequency /2 - programmable Fc
Bandwidth gain (Elliptic):	± 0.2dB max. (Fin ≤ 0.25 Fc)
Flatness (Elliptic):	± 0.2dB max.
Fc Gain :	-2 dB
2 x Fc Gain:	-60 dB (Elliptic) -25 dB (Pseudo-linear Phase)
3 x Fc Gain:	-55 dB (Elliptic) -60 dB (Pseudo-linear Phase)
Added Ripple :	20 Log [(THD + Bruit) / Vin_{FS}] = -70dB
Added Offset :	±10µV per kSamples/s
Added Offset Drift :	±10µV /°C max.
Added No Linearity :	± (0.01% FSR ± 1 mV)

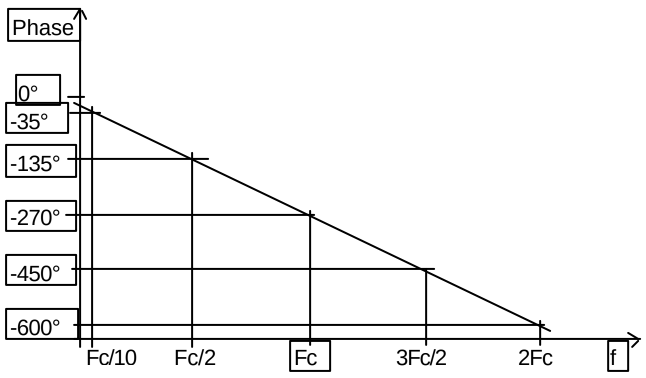
Elliptic Response :





Pseudo-linear Phase Response :





Option 06 characteristics

Number of Channels :	4
Output Modes :	2-wire or 4-wire mode
Isolation :	750VRMS & DC between channels and between channels to ground
Output Level (gain = 2) :	40 V_{pp} for 20 V_{pp} at the input (usable at 50V $_{pp}$ with G=2.5 & Iout<10mA)
Full Power Bandwidth :	DC à 20 kHz
Small Signal Bandwidth :	DC to 40 kHz (50kHz typical)
Slew Rate :	> 2V/µs
Settling Time :	50µs à 99.9% of the final value 350µs à 99.99% of the final value
Offset :	± 10mV à 23°C ± 2°C
Offset Drift :	± 1mV/°C typ., ± 5mV /°C max.
Gain Programming :	from 0 to 2.5
Gain Resolution :	6 .10^{-4} (12 bits)
Gain Accuracy :	± 15 mV
Gain Drift :	± 2 mV /°C
Absolute Non-linearity :	± 15 mV
Ripple Voltage (20 Hz to 20 MHz) :	15 mV RMS, 100 mV_{pp}
Channel to Channel Crosstalk :	< -60 dB at 10kHz
Input Impedance :	10 MΩ ± 5%
Input Protection :	250VRMS & DC
Output Impedance :	< 2 Ω in 2-wire mode < 0.2 Ω in 4-wire mode
Output Type :	Differential
Capacitive Load :	3.3nF max.
Maximum Output Current :	±50 mA per channel with gain ≤ 2 ±10mA per channel with gain > 2
Output Protection :	Permanent Short Circuit
User Connectors:	POSITRONIC SGMC 20, 26 pins

Option 06-S2 characteristics

Number of Channels :	4 or 8 (isolate output ground common for channels 1 and 2, 3 and 4, 5 and 6, 7 and 8)
Output Modes :	2-wire or 4-wire mode
Isolation :	750VRMS & DC between channels and between channels to ground
Output Level (gain = 2.5) :	80 V_{pp} for 20 V_{pp} at the input
Full Power Bandwidth :	DC à 20 kHz
Small Signal Bandwidth :	DC to 40 kHz (50kHz typical)
Slew Rate :	> 2V/µs
Settling Time :	50µs à 99.9% of the final value 350µs à 99.99% of the final value
Offset :	± 10mV à 23°C ± 2°C
Offset Drift :	± 1mV /°C typ., ± 5mV /°C max.
Gain Programming :	from 0 to 2.5
Permanent gain :	1.6
Gain Resolution :	$6 .10^{-4}$ (12 bits)
Gain Accuracy :	± 15 mV
Gain Drift :	± 2 mV /°C
Absolute Non-linearity :	± 15 mV

S3 option for 6062 module

Aim

The aim of S3 option is to acquire or generate a signal at the same sampling frequency as external clock. Specific drivers are available for Option S3.

Using with external clock

External clock frequency range is from 0.1Hz to 1MHz.
Sampling occurs on fall edge.

In order to program the sample frequency, use the following formula, and send “Value” using shared-resource register mode to the adequate channel.

Fsample = 2 x Fext / Value.

When “Value” is equal to 2, sampling frequency is equal to external clock frequency.

Value range is from 2 to 1000000.

Using with 10MHz internal clock

In order to program the sample frequency, use the following formula, and send “Value” using shared-resource register mode to the adequate channel.

Fsample = 2 x 10MHz/ Value.

For analog input board, “Value” range is from 200 to 1000000 (100kHz to 20Hz).
For 250kS/s analog input board, “Value” range is from 80 to 1000000 (250kHz to 20Hz).

For analog output board, “Value” range is from 40 to 1000000 (500kHz to 20Hz).
For 2.5MS/s analog output board, “Value” range is from 8 to 1000000 (500kHz to 20Hz).

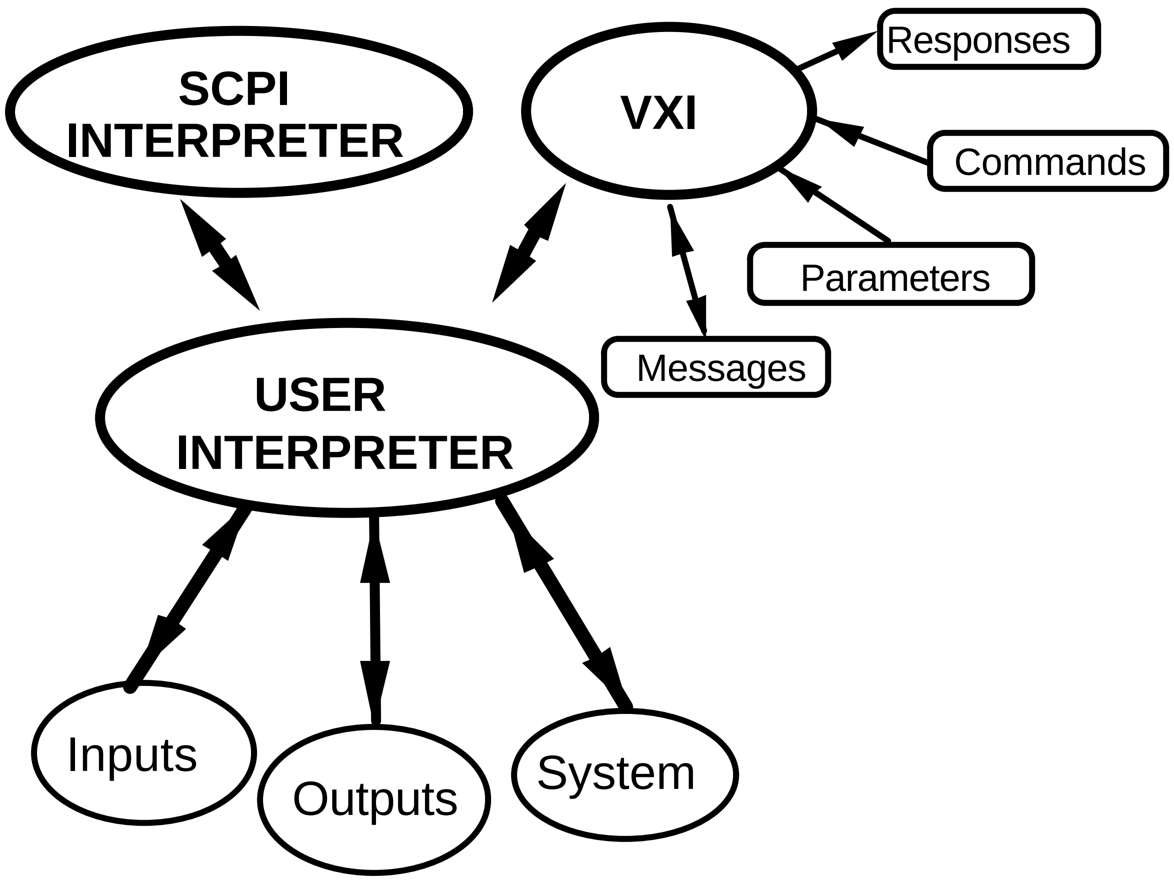
Recommendations

Never use TRIG :TIM command cause of software protection. S3 sampling time bases are managed differently than standard sampling time bases. Therefore using this command make your software false.

Never program a sampling frequency highest than 100kHz for Analog Input, 250kHz for 250KS/s Analog Input, 500KHz for Analog output and 1MHz for 2.5MS/s Analog output channels.

Functional Description

Internal Software Organization



USER INTERPRETER: this is the main module, which manages the function calls coming from other modules

VXI: controls the basic and advanced functions for reading from and writing to the VXI bus.

SCPI INTERPRETER: this interpretation module translates messages coming from the VXI bus into function calls or error messages.

Inputs and Outputs: control the input/output functions.

System: manages the principal system modules (DMA, time base, clock and voltage reference, etc.)

Electrical description

The 6062 module includes four functional groups:
- signal group (LED's)
- analog input group
- reference input/output group
- analog output group

Signal Group

This group includes three signal lamps, which serve the following purposes:

Fail: Indicates the module is not operational. This LED remains lit during module initialization upon power-up (5 seconds maximum).

Access: A very brief flash indicates the module's command interpreter has received a command.

Test: Remains lit during the self-test sequence either when automatically started (during initialization) or when a command is received for a self-test.

Analog Input Group

The four analog inputs are accessible via the **J1**connecter**.**

Input and output triggers are accessible via the **J2** connector.

Reference Input/Output Group

The **J3** connector provides input for an external reference voltage in order to test the module compared to a standard voltage.

The **J4** connector provides an output of the internal reference voltage of 10.000 V (see the general specifications).

The **J5** connector provides for the use of an alternate reference clock (common to all channels) to obtain lower sampling frequencies than those available as standard (1 sample per second minimum). Refer to the general specification for conditions of use.

Analog Output Group

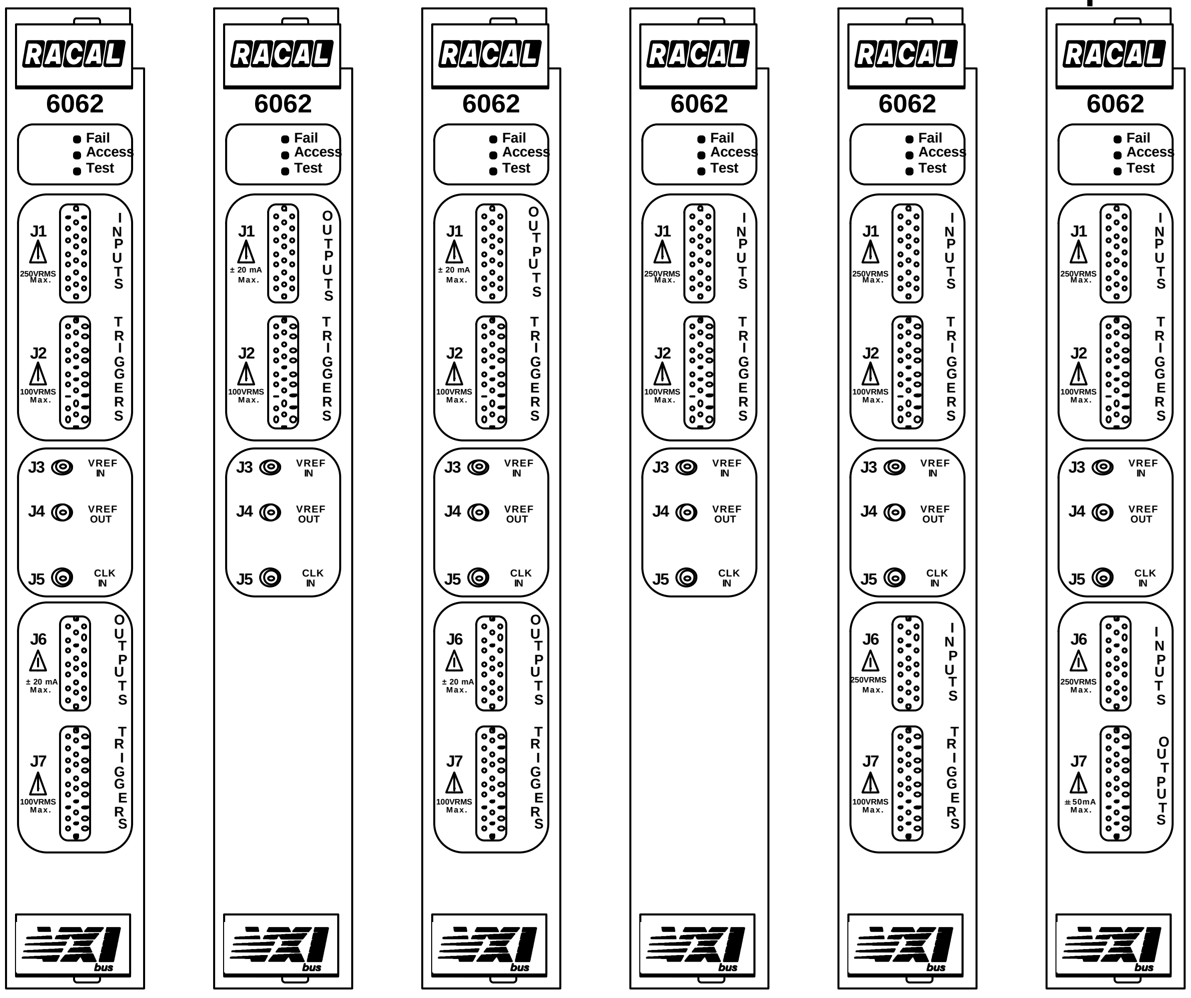
The four analog outputs are accessible via the J1 connector.

Input and output triggers are accessible via the **J2** connector.

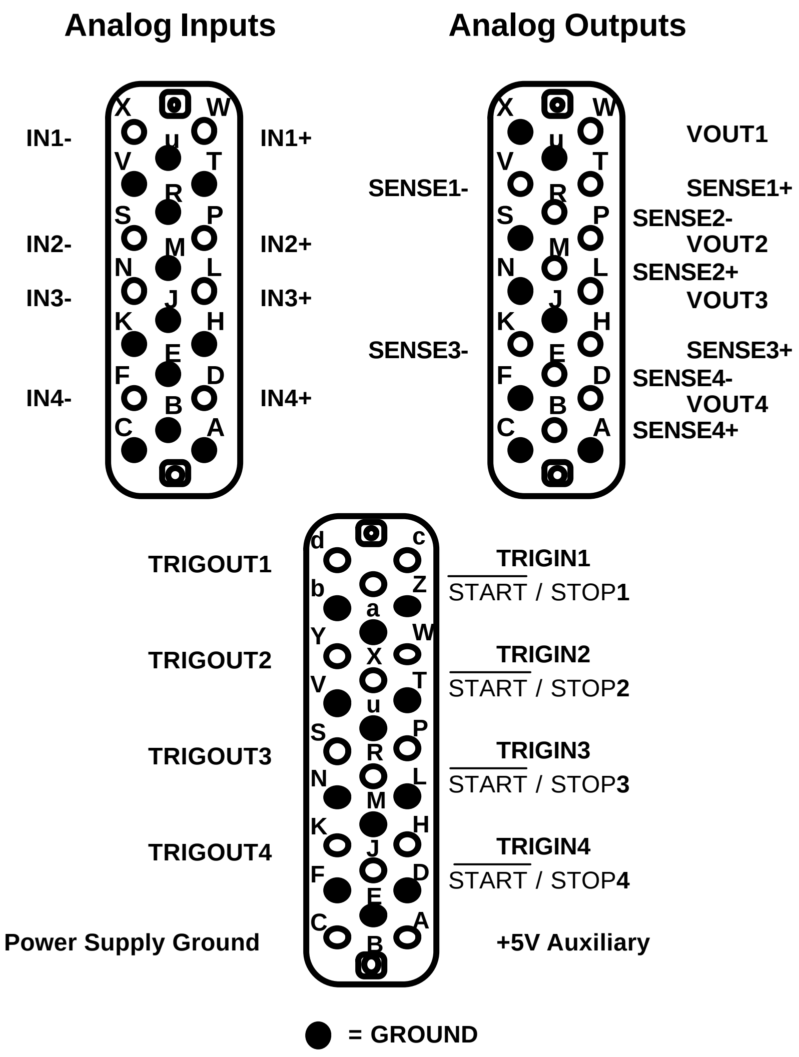
Note The connectors corresponding to these input/output channels are provided with the device. It is recommended to use only the type of connectors provided.

Physical Description

6062A **6062B4** **6062B8** **6062C4** **6062C8** **6062B4 Opt 06**

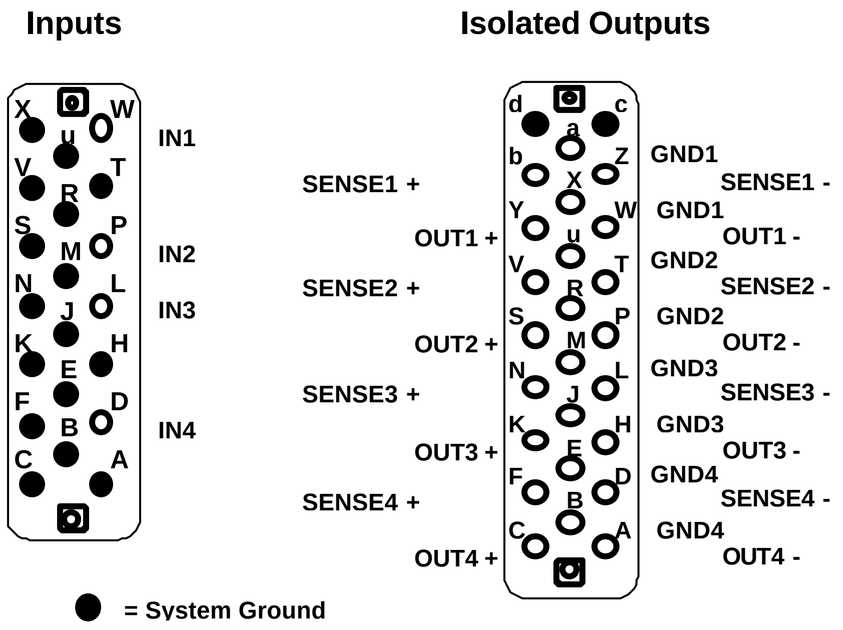


User Connectors (Front Panel)

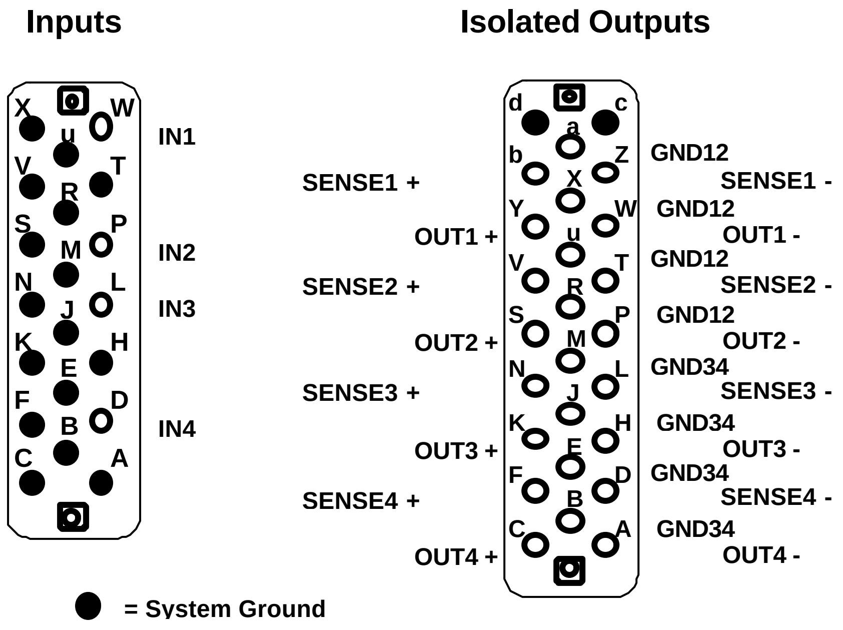


Input/Output Triggers

Option 06 Connectors (Front Panel)



Option 06-S2 Connectors (Front Panel)



2. Configuring the Instrument

Installation

Configuring the logical address

The 6062 may be installed in any available slot in a size C VXI bus-compatible chassis except for the one farthest to the left which, is reserved for the controller.

A free address is required to use the 6062 module. The logical address is configured, when the power is off, using the switches located on the upper side, near the VXI bus connectors. The switches are used for static or dynamic configuration. Values from 1 to 254 may be used for static configuration.

An alternate method for configuring the instrument is via dynamic configuration (see section F of the VXI bus system specification, revision 1.4). To operate in this mode, the user must set all the switches to the ON position (255).

The position corresponding to address 0 (all switches set to OFF) is reserved.

Interrupt number 1 corresponds to the least-significant bit.

Note ON corresponds to a logical 1, and conversely OFF corresponds to a logical 0.

Configuring the address of shared resources

During the module's initialization phase, the VXI bus controller dynamically allocates an address to the shared resources of the 6062 module.

This allocation concerns only the upper eight bits of the base address of the shared resources (A24 to A31). This base address will be used to directly address the shared resources after having made a request to share the bus via the associated protocol.

Configuring the VXI Bus Interrupts

One of the seven available interrupt lines is programmable on the 6062 module. This line is assigned using the word serial commands reserved for interrupts, and results in setting the int_ID bit to 1.

Installing the 6062 in a 9; , chassis

To install a 6062 module in a size C VXI chassis, make sure the power to the chassis has been switched off.

Correctly configure the interrupt acknowledge lines at the bottom of the VXI chassis in a daisy chain, so the interrupt travels across the free slots (see the VXI bus specifications).

Remove the chassis cover and insert the 6062 in the appropriate slot, so that the signal lamps are positioned towards the top of the chassis (or towards the left for a horizontal chassis).

Initialization after power-on

Before switching on the power to the chassis, make sure the bus controller (slot 0) is present.

The following steps occur during initialization:

- Control of the card identifier
- Interface initialization
- Power on self-test: test is run on ROM, RAM, FLASH EEPROM, the micro-controller and all the logical components attached to it.
- Disable the TTLTRIG lines
- Analog input/output interface self-test

Self-test

The self-test is run upon a user request via an appropriate command. The tests concern the logical functions of the 6062 module. The tests are run in less than five seconds, as recommended in the bus specifications. It is not necessary to use external test equipment or special cabling to execute these tests.

Note During the test phase, analog inputs and outputs to the test module are disconnected from use (high impedance). Upon completion of the test, they are automatically reset to their state preceding the test.

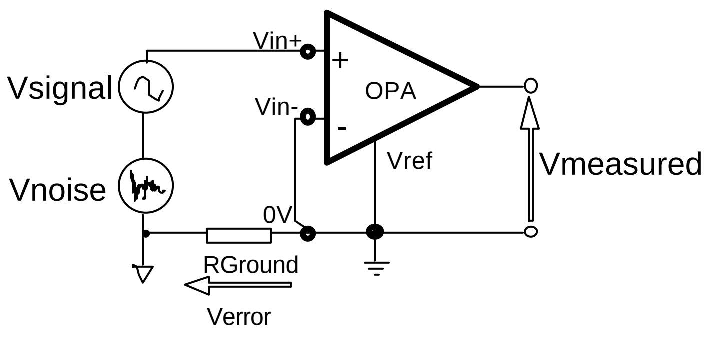
3. Using the instrument

Introduction

For clarity's sake, the following description concerns the 6062A module (4 analog inputs, 4 analog outputs). It is simple to extrapolate this information to the 6062B and 6062C models, since the 6062A includes all the features of the other two modules.

Using the analog inputs (6062A and 6062C-X)

The analog inputs are differential, which provides better noise reduction on the signal to be measured. It is nonetheless possible to use inputs in simple termination mode. The figures below show the relative performance of the analog inputs in the two modes mentioned previously.

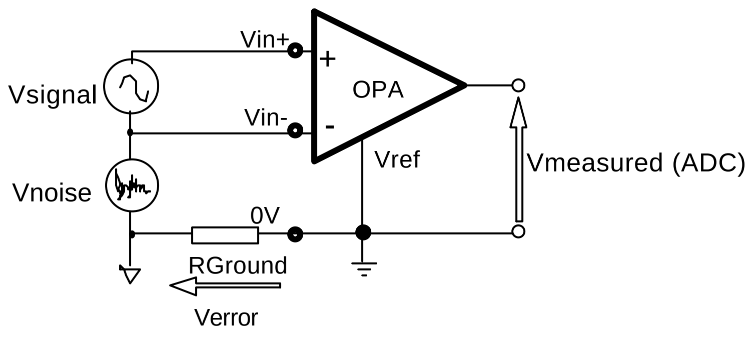


In the first case (simple termination mode), the voltage measured by the analog/digital converter (ADC) equals:

$$V_{measured} = G\,(V_{signal} + V_{noise}) + (G-1)\,V_{error}$$

Where **G** (differential gain) is approximately equal to 1, if R_{ground} is negligible compared to the impedance of the differential input.

The second figure displays how to connect the differential input to the source to be measured, to improve performance:

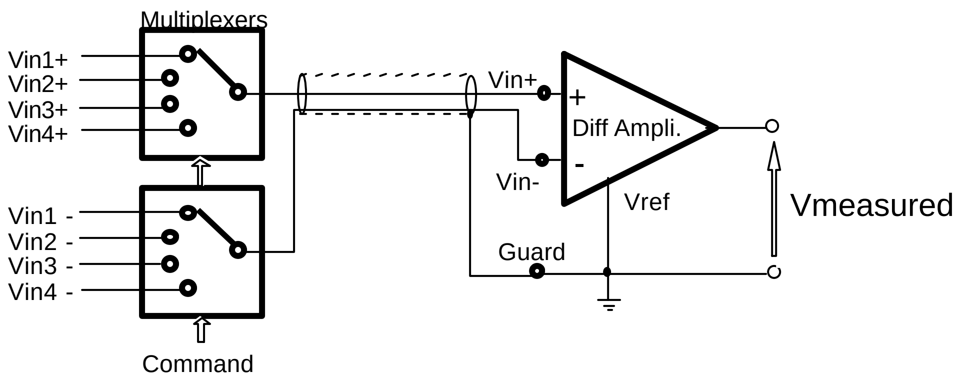


With this type of interconnection, the differential non-galvanically isolates the source of the reference earth. The differential input stage amplifies the low-level differential signals while rejecting signals that are common to both inputs (common mode voltage).

The voltage measured by the analog/digital converter in this case is:

$$V_{measured} = G\,(V_{signal})$$

The use of input multiplexed requires a few cabling precautions in differential mode. The figure below shows one method for connecting the multiplexer with the differential input:

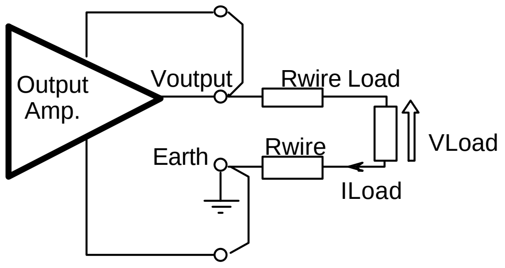


Using the Analog Outputs (6062A and 6062B-X)

The analog outputs offer two operating modes:

- Simple termination mode
- "4-wire" mode

Simple termination mode may be used when the current applied to the output remains low. When current increases, the resistance of the output wires on the load, and the resulting voltage error, become significant.



The figure above presents an example of how to connect the Load in simple termination mode. In this case, the available voltage across the terminals is:

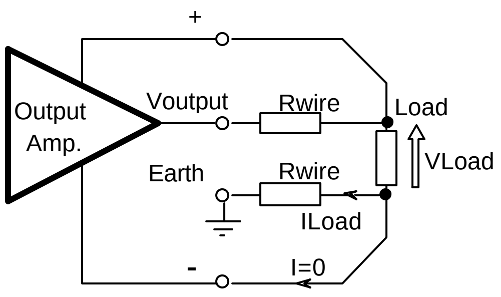
$$V_{charge} = V_{output} - 2\,(R_{wire} \times ILoad)$$

If we take as an example a typical case where:

- Voutput = 10V
- ILoad = 20mA
- Rwire = 200mΩ

The voltage actually available across the terminals is 9.992V, (error = 8mV).

Using the following connection diagram, the voltage is measured at the terminals using the + and - inputs, which provides a warning at the output level, so that VLoad is approximately equal to Voutput.

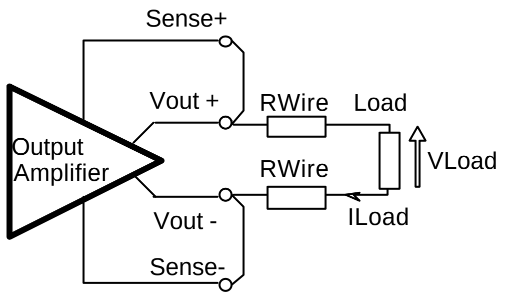


Using the Isolated Outputs (6062B4-Opt06)

The analog outputs offer two operating modes:

- Simple termination mode
- "4-wire" mode

Simple termination mode may be used when the current applied to the output remains low. When current increases, the resistance of the output wires on the load, and the resulting voltage error, become significant.



The figure above presents an example of how to connect the Load in simple termination mode. In this case, the available voltage across the terminals is:

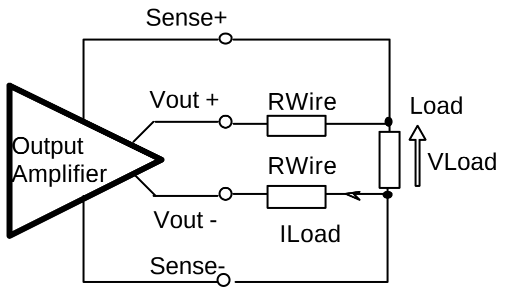
$$V_{charge} = V_{output} - 2\,(R_{wire} \times ILoad)$$

If we take the previous example where:

• Voutput = 10V
• ILoad = 20mA
• Rwire = 200mΩ

The voltage actually available across the terminals is 9.992V, (error = 8mV).

Using the following connection diagram, the voltage is measured at the terminals using the + and - inputs, which provides a warning at the output level, so that VLoad is approximately equal to Voutput.



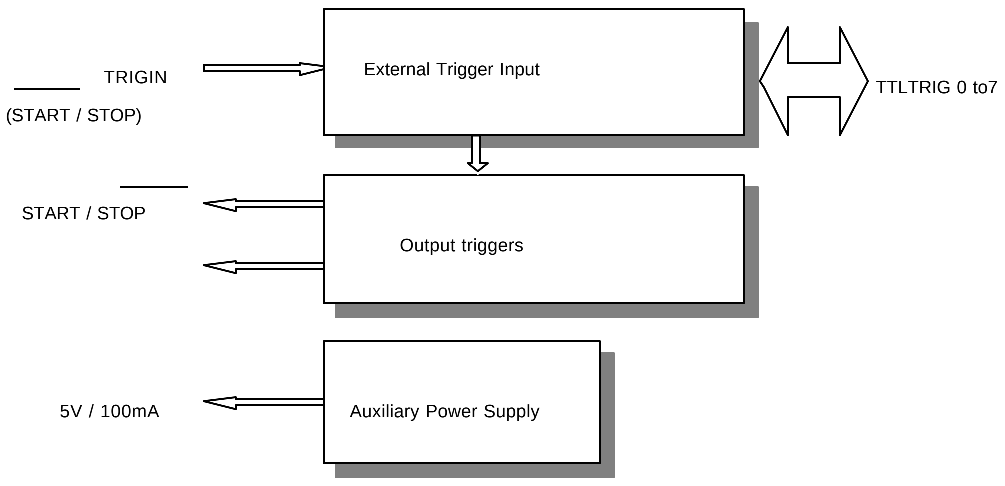
Triggering the instrument

Internal Trigger

Each channel may be triggered by an SCPI command, which has priority over an external trigger (see Chapter 5).

External Trigger

Three distinct functional blocks allow the user to trigger the instrument by external events. These events may come from a selected TTLTRIG line, or from the external trigger input (TRIGIN). The following figure illustrates the functional division of the blocks associated to each Input/Output channel:



The signals on the left of the diagram represent the Input/Output channels available on the J2 and J7 trigger connectors (TRIGGERS).

The TRIGIN input represents a dual state START/STOP triggering line used to start a generation or an acquisition on the considered channel. The TTL signal can trigger the channel when it presents a 0 state. The logical level 1 of this signal determines the inactive state.

Note When no signal is connected to TRIGIN, the channel is in the STOP state (logical level 1)

The TRIGIN input may be configured to command one of the eight available TTLTRIG lines. In this case, the line may not be used to trigger that channel.

With the Option 10 (edge triggering), the module can be triggered by a negative or positive pulse but, the effective edge took in reference will be the rising edge.

Using external references

10.000V output reference (J4)

This output is the reference image for the internal 10V voltage level. It is used during the self-test sequences for the analog input/output channels.
It may also be used as a reference for specific external conditioning. Remember that the current is limited to 20mA for this output.

10.000V input reference (J3)

This input is used during the self-test sequences for the analog input/output channels. This reference should not be used during the sequence, as that would create a self-test error.

Reference clock input (J5)

This input may be used to provide a time base reference for the eight analog channels for sampling periods exceeding one second (to analyze slow phenomena). A minimal clock frequency of 10Hz can accept sampling periods of more than 27 hours (0.1s x 1,000,000).

Saving the configuration

The 6062 module includes a bank of non-volatile "Flash" memory that can keep vital system data (not accessible to the user), and sampling tables. These tables may be used to generate base signals such as:

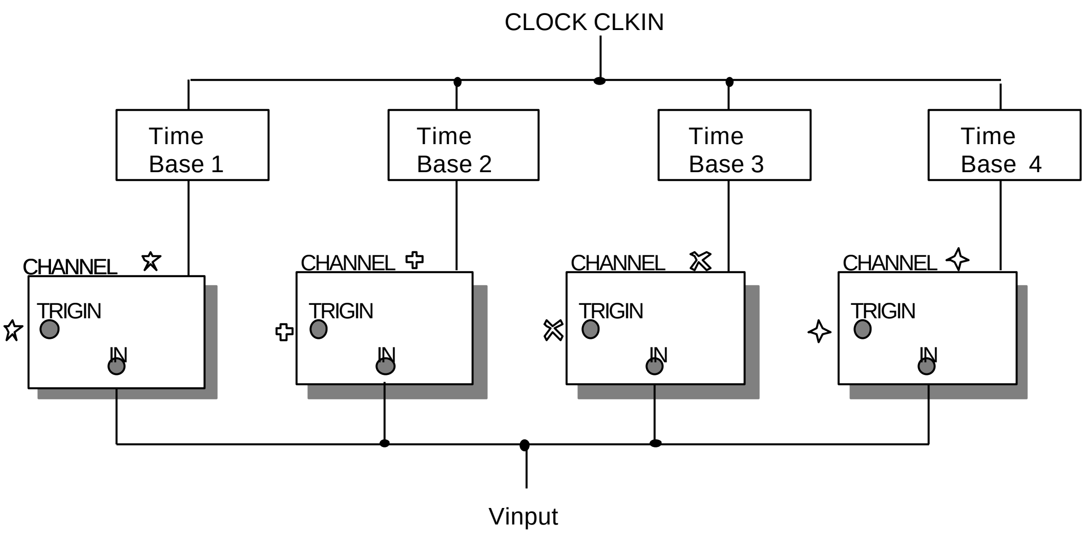
- Slope
- Triangle
- Sine
- Square
- Arbitrary signals
- Etc.

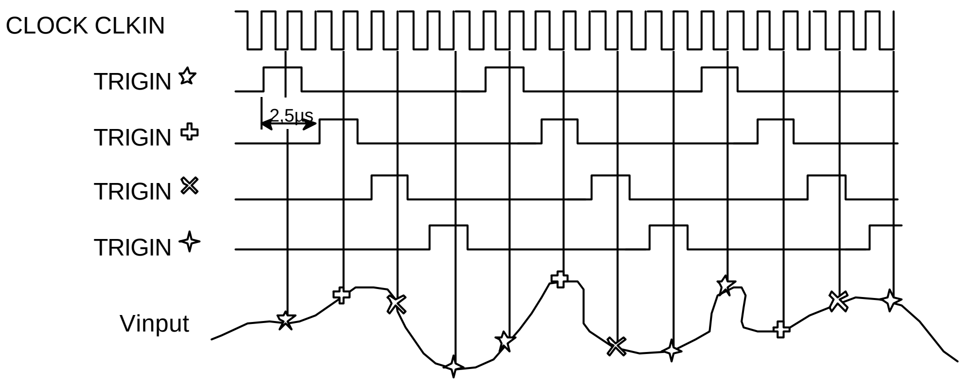
See Chapter 5 for commands to access this non-volatile memory.

Using the option 02 (400kS/s)

1- Measure a channel at 400,000 samples/second

Hardware Installation





The figure above displays the connections required to operate in this mode. Among other things, the input filter must be disconnected in order to increase the bandwidth of the four analog channels.

The clock used in this case must be supplied to the module via the CLKIN input on the module's front panel (J5). The Option 02 provides the synchronization for each input channel (TRIG IN), in order to trigger each channel at 2.5 µs intervals.

Note The Option 05 (1MS/ s) follows the same principle but requires the Option 07 (Four or eight input channels at 250KS/s) to run properly.

Software Installation

The following list of SCPI commands is used to configure the module in this mode.

Module Initialization

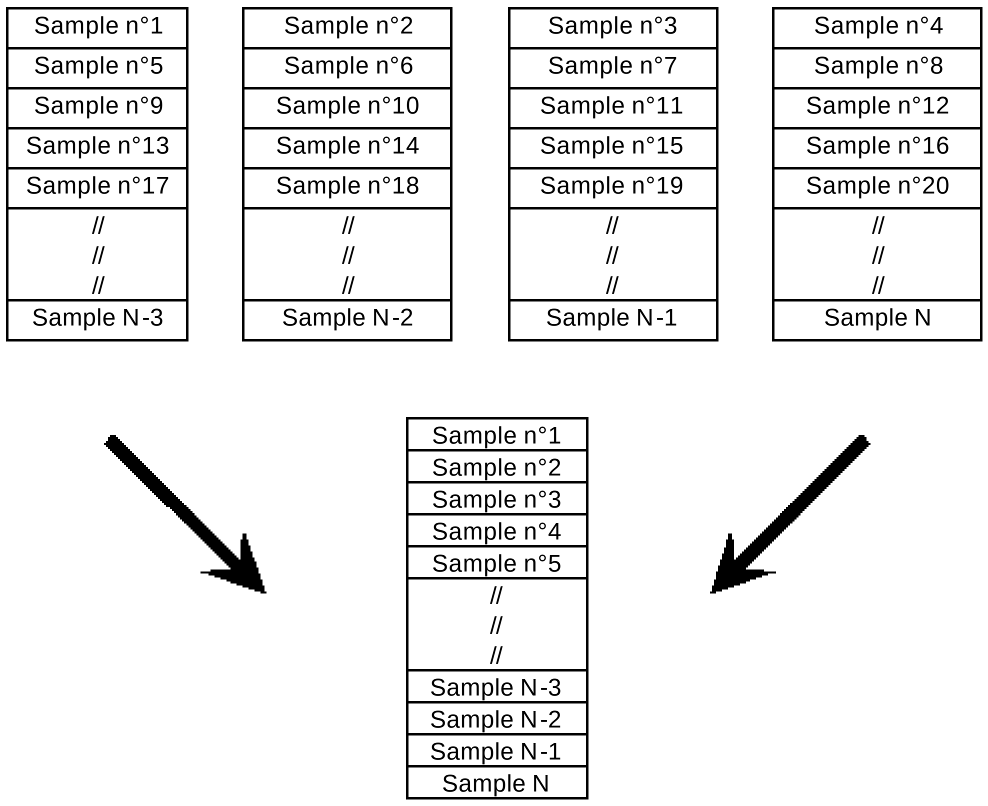
Commands	Result
SYST:CLOC EXT	External frequency reference
TRIG:TIM 10, (@1:4)	Sample at 100kSamples/sec.
TRIG:STAR:SOUR ETRG, (@1:4)	External trigger
OUTP ON (@1:4)	Close output relays
INIT (@1:4)	Wait for external trigger

In this case, consecutive samples are divided into four memory blocks associated with each channel. In order to rebuild the input signal the user must recover the samples sequentially in each of the four memory blocks.

With the Option 05 (1MS/s) the SCPI commands are :

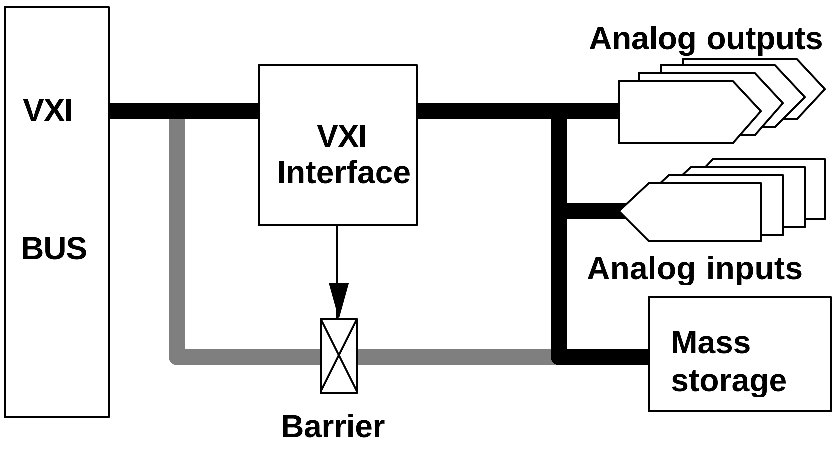
Commands	Result
SYST:CLOC EXT	External frequency reference
TRIG:TIM 40, (@1:4)	Sample at 250kSamples/sec.
TRIG:STAR:SOUR ETRG, (@1:4)	External trigger
OUTP ON (@1:4)	Close output relays
INIT (@1:4)	Wait for external trigger

The figure below illustrates how samples are stored in the memories of the four channels used, as well as the order in which they should be retrieved to restore the original signal.



4. Instrument Operation

Description of the 6062 module



The preceding synoptic schematic illustrates the different functional blocks that make up the 6062 module.

- **The VXI bus** carries messages and data from (to) the VXI interface.

- **The VXI interface** provides the link between the various elements of the module and the user via the VXI bus.

- **The resources** include the input/output channels and mass storage.

- **The tri-state barrier** lets the user access resources directly from the VXI bus.

The two operating modes co-exist:

- The message-based mode used to communicate with the module via high-level commands.

- The shared resource mode used to exchange blocks of data directly with the module, thus increasing the data transfer rate.

Message-based mode

At power-on, the module is automatically configured to work in message-based mode. Resources are not "visible" except through the word serial protocol.
To enable direct access to the resources, a request to share bus resources must be made respecting the associated protocol.
When the module has replied favorably to a bus request, the resources are available to the user.

The flexibility and ease-of-use associated with message-based mode are especially useful during configuration sequences, or to send complex commands to many entities of the module.

On the other hand, the major drawback of this method of communication is the comparatively slow exchange rate compared to the direct memory block transfer. Furthermore, the delay for interpreting messages varies according to the priority of the tasks being executed by the microprocessor of the VXI interface. This mode should not be used for operations where the predictability of events following the interpretation of a message is critical.

Refer to Chapter 5 for details on controlling the instrument in this mode.

Shared Resource mode

Warning Since the "Shared resource" mode uses very specific data exchange rules via the intermediary of the VXI bus, users should be thoroughly familiar with the bus specification before pursuing this chapter.

As opposed to the message-based mode, this mode provides considerably higher sample transfer rates for data capture and generation. It provides better control for certain trigger operations and for more complex operations requiring many message-based commands or for which no commands exist.

Operating mode

To directly access resources, the bus controller must first request to share the 6062's resources. This is a **handshaking** protocol, specific to this module. It is highly recommended to respect the unfolding of the operations associated with this protocol, or else risk serious module dysfunction.

Chronology

1) The VXI bus controller (slot 0) writes the value **0001_{hex}** in the shared resource register. The physical address of the shared resource register is **$YYYY_{hex}$ + 20_{hex}**, where the YYYY value depends on the logical address defined by SW1, as follows:

$C000_{hex}$ + (40_{hex} x Logical Address in hexadecimal)

Or in decimal:

49152 + (64 x Logical Address)

Example: if the logical address is 24, the address of the shared resource register is $50720_{decimal}$ or $C620_{hex}$.

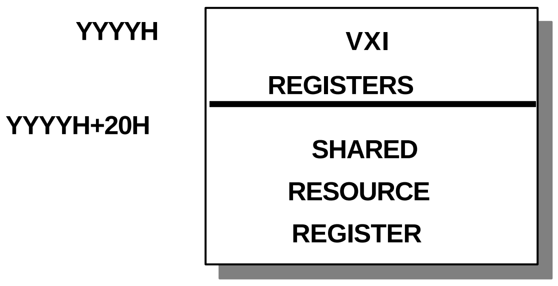
2) The microprocessor terminates the current instruction, then the module sends the value **0001_{hex}** via the same shared address register (address **$YYYY_{hex}$ + 20_{hex}**) to accept the share request.

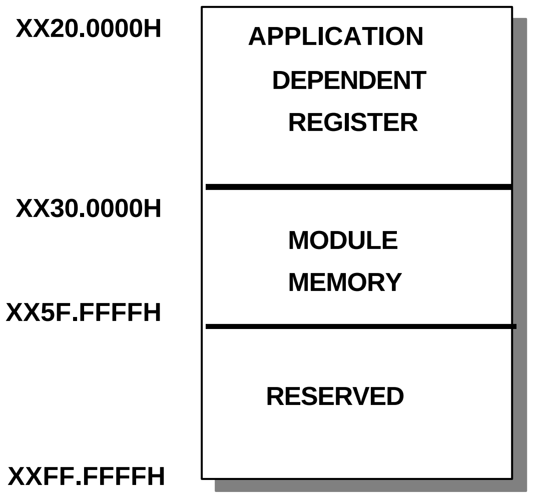
3) As soon as the VXI bus controller receives the value **0001_{hex}** , data exchanges may begin between the bus controller and the module.
During the entire exchange, the module acts like memory segmented into many blocks associated to the slave decoding logic of the VXI bus.

4) The VXI bus controller (slot 0) writes the value **0000_{hex}** in the shared register to terminate the shared resource protocol.

Note During the shared resource phase, message handling is interrupted. It is necessary to "give back the bus" to the module to recommence message-based communications.

The figure below illustrates the two address areas, first for the VXI registers and the shared resource registers, then for application-dependent registers and the module memory area.





VXI Registers

These registers conform to the VXI bus specification, revision 1.4. Refer to this document for details.

Shared-resource register

Only the D00 bit is significant in this register. The bus controller sets the bit to 1 during a write cycle to signify a request to share the bus. The controller must wait for this bit to be set (read cycle) to share the module's resources.

Write mode to the address YYYYH+20H

D15															D00
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	1

Bus sharing request bit ↗

Read mode from the address YYYYH+20H

D15															D00
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	1

Bus sharing accept bit ↗

Application-dependent registers

These registers, an integral part of the module's resources, configure and control each input/output channel. The following table shows the different types of registers, their addresses and significance:

R/W	REGISTER	CHAN.1	CHAN. 2	CHAN. 3	CHAN. 4	CHAN. 5	CHAN. 6	CHAN. 7	CHAN. 8
W	Start Soft	200000	201000	202000	203000	208000	209000	20A000	20B000
W	Stop Soft	200002	201002	202002	203002	208002	209002	20A002	20B002
W	Freq. -HI	200006	201006	202006	203006	208006	209006	20A006	20B006
W	Freq. -LO	20000A	20100A	20200A	20300A	20800A	20900A	20A00A	20B00A
W	LEN	20000C	20100C	20200C	20300C	20800C	20900C	20A00C	20B00C
W	Count	20000E	20100E	20200E	20300E	20800E	20900E	20A00E	20B00E
W	ADC					208018	209018	20A018	20B018
R	RAM input channel	400000 to 41FFFF	420000 to 43FFFF	440000 to 45FFFF	460000 to 47FFFF				
W	RAM output channel					500000 to 51FFFF	520000 to 53FFFF	540000 to 55FFFF	560000 to 57FFFF

6062B4 Option 06 Gain Configuration

R/W	REGISTER	CHAN.1	CHAN. 2	CHAN. 3	CHAN. 4	Isolated Out 1	Isolated Out 2	Isolated Out 3	Isolated Out 4
W	Gain Control	-	-	-	-	208000	208002	208004	208006

Configuration and channel memory registers

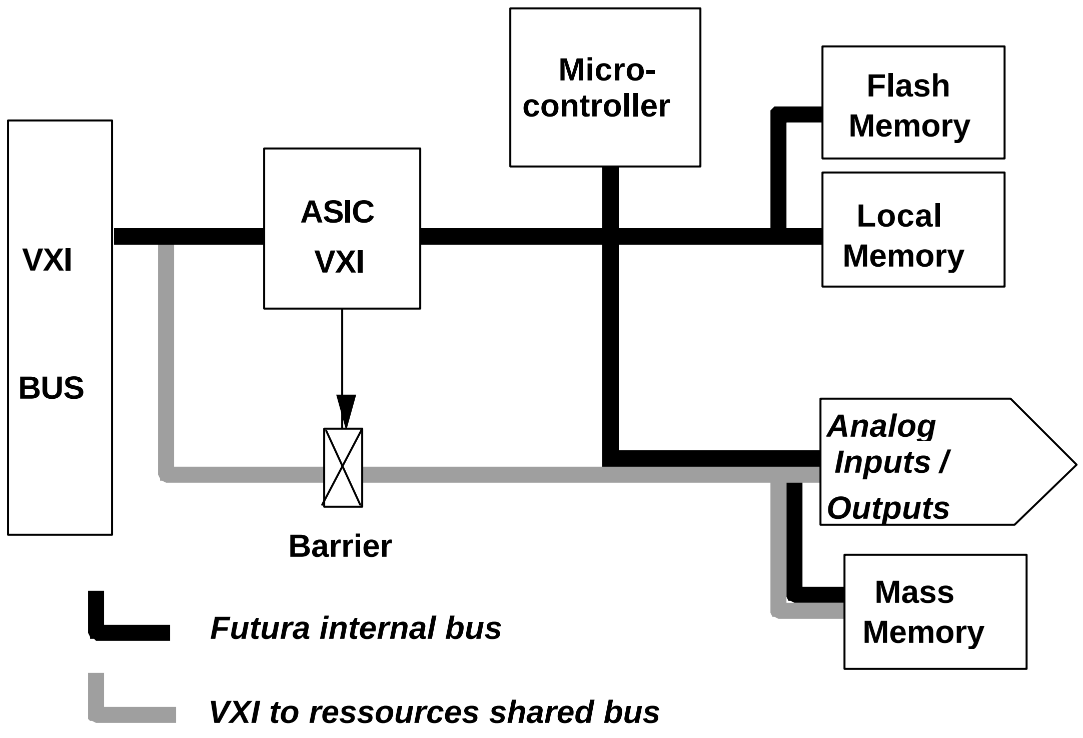
R/W	REGISTER	SIGNIFICANCE	HEX ADDRESS
W	CSRELAY	Controls analog output channel switches	201020
W	CSTRIG	Configures TTLTRIG lines	206XXX
W	CSIRQ	Authorizes control TTLTRIG and TRIGIN Lines	200020

Relays and Trigger configuration registers

The data field is interpreted as follows:

REGISTER	INTERPRETATION OF DATA FIELD
Start Soft	No significance
Stop Soft	No significance
Freq.-HI.	4 high-order bits of the period (µs)
Freq.-LO	16 low-order bits of the period (µs)
LEN	Number of 16-bit samples
Count	Number of sampling cycles
DAC	16-bit value sent directly by the Digital/Analog Converter 0000_{hex} for 0V output $7FFF_{hex}$ for +9.9997V output 8000_{hex} for -10.000V output
ADC	16-bit value read directly from the Analog/Digital Converter 0000_{hex} for 0V input $7FFF_{hex}$ for +9.9997V input 8000_{hex} for -10.000V input
RAM Channel	16-bit words
CSRELAY	Bit D0 = 1 to validate the selected TTLTRIG Line on channel 1 Bit D1 = 1 to validate the selected TTLTRIG Line on channel 2 Bit D2 = 1 to validate the selected TTLTRIG Line on channel 3 Bit D3 = 1 to validate the selected TTLTRIG Line on channel 4 Bit D4 = 1 to control the relay on channel 1 Bit D5 = 1 to control the relay on channel 2 Bit D6 = 1 to control the relay on channel 3 Bit D7 = 1 to control the relay on channel 4
CSTRIG	Bits D0 to D2 to select a TTLTRIG line for channel 1 Bits D4 to D6 to select a TTLTRIG line for channel 2 Bits D8 to D10 to select a TTLTRIG line for channel 3 Bits D12 to D14 to select a TTLTRIG line for channel 4 Bit D3 = 1 to configure the selected TTLTRIG Line on channel 1 as an input Bit D7 = 1 to configure the selected TTLTRIG Line on channel 2 as an input Bit D11 = 1 to configure the selected TTLTRIG Line on channel 3 as an input Bit D15 = 1 to configure the selected TTLTRIG Line on channel 4 as an input
CSIRQ	Bits D0 & D1 to select an event as an IRQ for the channel 1 Bits D2 & D3 to select an event as an IRQ for the channel 2 Bits D4 & D5 to select an event as an IRQ for the channel 3 Bits D6 & D7 to select an event as an IRQ for the channel 4 Bit D8 to validate the selected event for the channel 1 Bit D9 to validate the selected event for the channel 2 Bit D10 to validate the selected event for the channel 3 Bit D11 to validate the selected event for the channel 4 Bit D12 to validate a TRIGIN line on channel 1 Bit D13 to validate a TRIGIN line on channel 2 Bit D14 to validate a TRIGIN line on channel 3 Bit D15 to validate a TRIGIN line on channel 4
Gain Control (Option 06)	Bits D0 to D11 to configure the gain on each channel. $X000_H$ = Gain 0; $XFFF_H$ = Gain 2.5

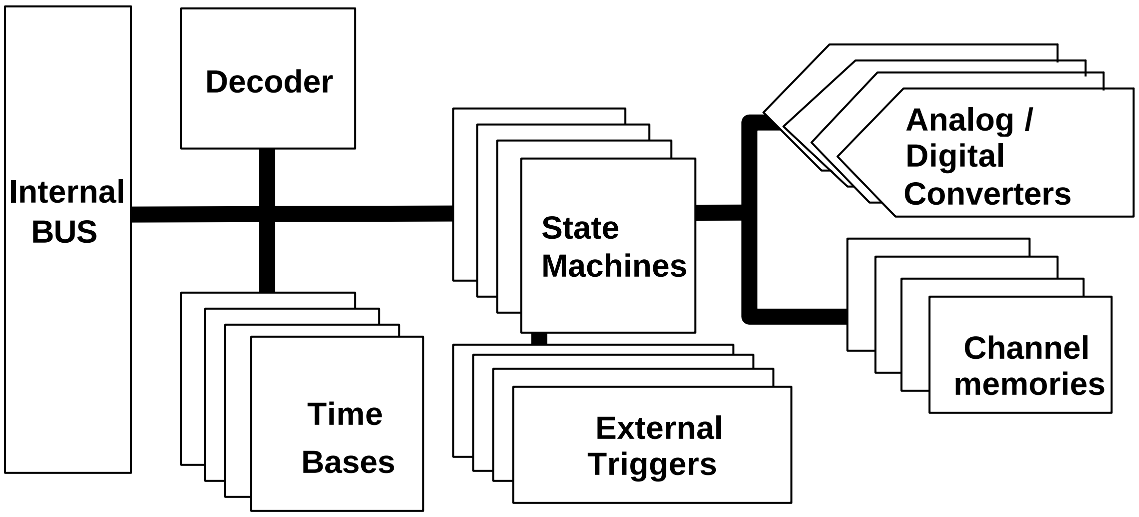
VXI Interface Description



The VXI interface includes the elements necessary to execute the following functions:

- VXI registers
- Message-based command interpreter
- Bus sharing for direct resource access
- Control of interrupt lines
- Control of TTLTRIG triggers

Analog Input Description



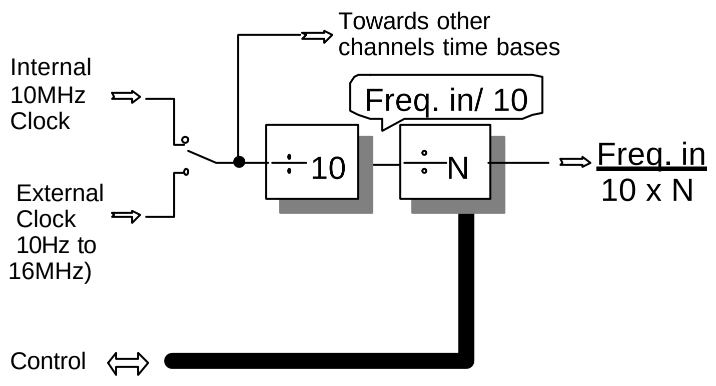
All the analog input channels are located on an application card, which in turn is connected to one of the two application ports on the motherboard. This architecture provides greater flexibility when choosing the options for the 6062 module.
The analog input channels communicate with the VXI interface via an internal bus which may be directly shared with the VXI bus (shared resource mode).
Each channel is independent and thus has available:

- a sampling time base
- a state machine
- an Analog to Digital converter
- a sampling memory
- a trigger signal multiplexer

All of these units may be controlled via the decoder which manages the link between the different addressing areas listed above (pages 4-5 and 4-6) and the associated resources.

Detailed functional block description

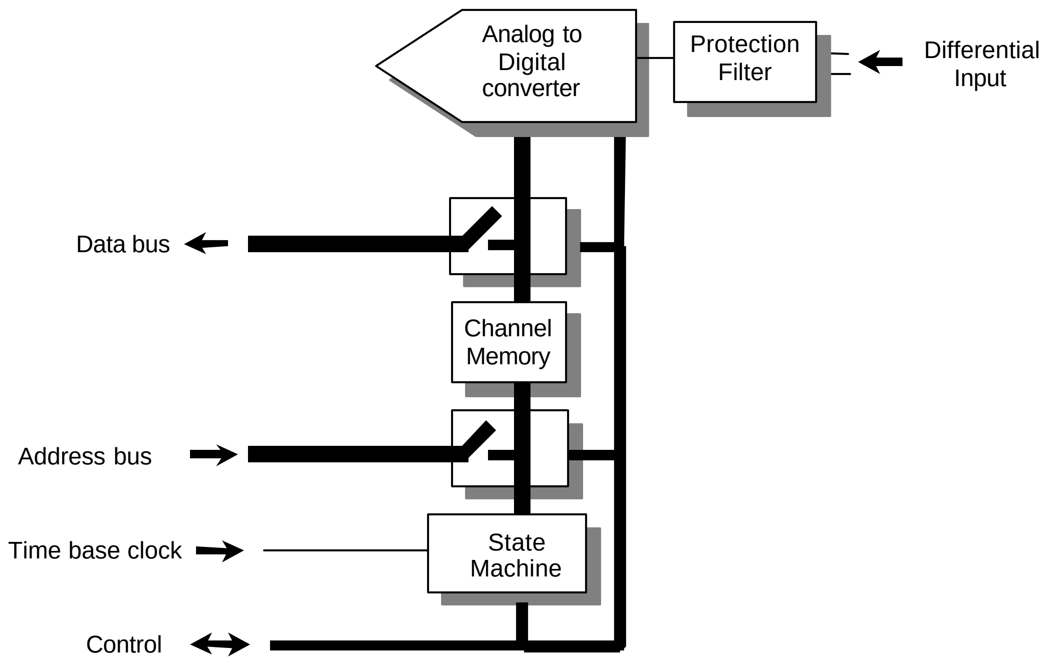
Sampling time bases


Towards other channels time bases
Internal 10MHz Clock
External Clock 10Hz to 16MHz)
Freq. in/ 10
÷10
÷N
Freq. in / 10 x N
Control

The programmable time base is made up of a fixed divisor of 10, plus a programmable divisor whose value (N) may be programmed from 10 to 1,000,000. On power-up, the internal clock used is a 10MHz clock from the VXI bus (ECLCLK). The extreme programmable sampling values in this case are from 1Hz to 100kHz, with a resolution of 1 microsecond.

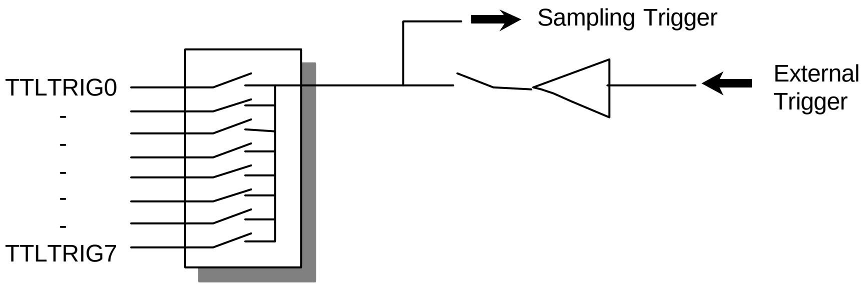
Note The chosen clock frequency (internal or external) is applied to all input/output channels, while each time base is independently programmable (N).

State machine, converter and sampling memory



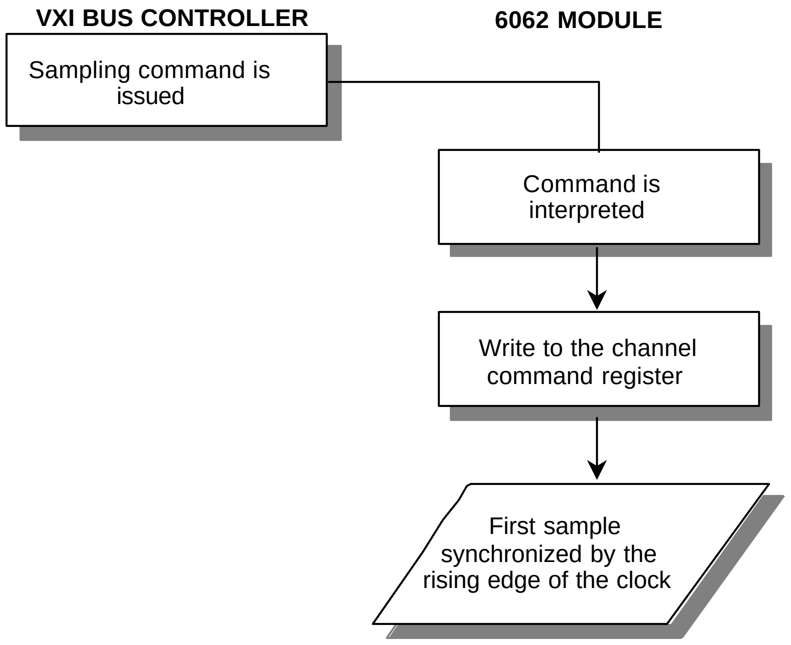
The state machine brings samples from the analog/digital converter to the channel memory and makes them available on the data bus during the conversion phases. The programmable time-base clock times the conversion/memory write cycles.

Trigger signal multiplexer

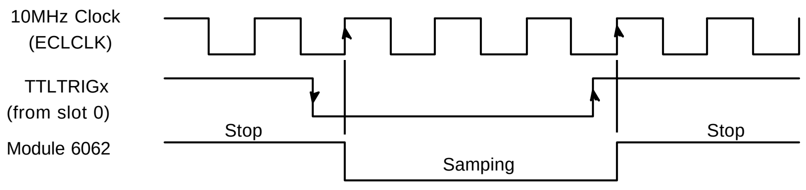


Sampling may be triggered by different events such as:

- *software trigger*: sampling begins when the corresponding command has been received and interpreted (non-synchronous triggering unless the MODE SYNC ON command has been sent previously).

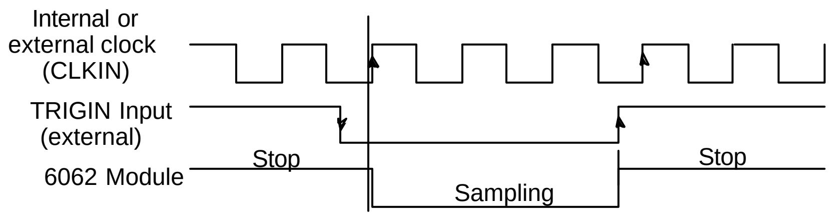

VXI BUS CONTROLLER
6062 MODULE
Sampling command is issued
Command is interpreted
Write to the channel command register
First sample synchronized by the rising edge of the clock

- *trigger by activation of one of the eight TTLTRIG lines* (START/STOP protocol): the selected TTLTRIG line and the ECLCLK clock of the VXI bus control The sampling cycle. The following timing diagram describes the protocol:

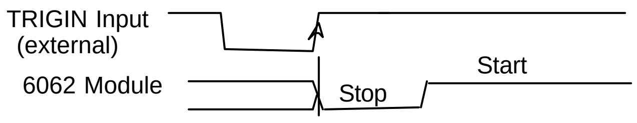


Note The START/STOP protocol requires the use of the ECLCLK clock as a synchronization reference (default configuration).

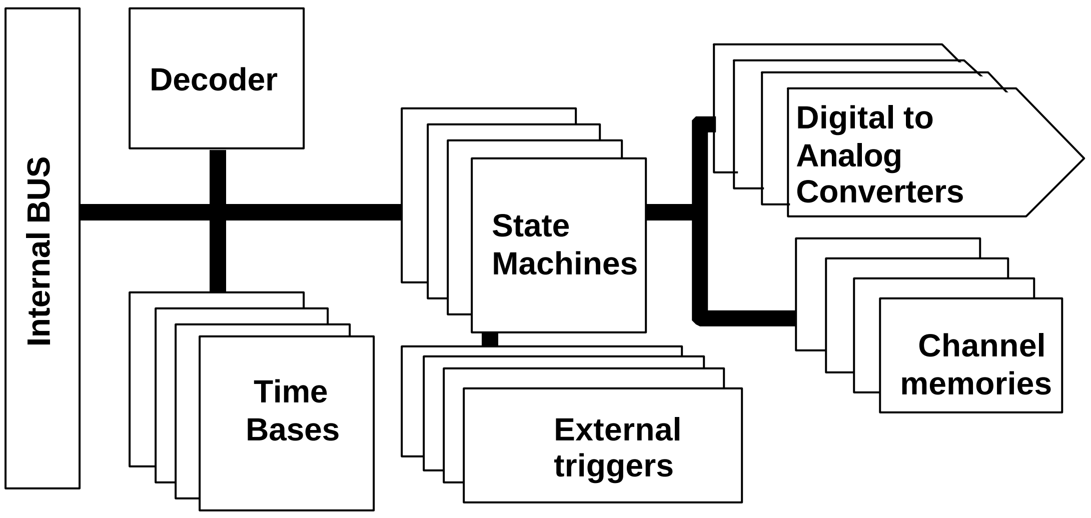
- *external trigger*: The sampling cycle is controlled by the activation of the TRIGIN input (J2). The first sample is synchronized on the rising edge of the selected clock, and terminated by the appearance of a rising edge on the TRIGIN line.



- *external trigger with Option 10*: The sampling cycle is controlled by the edges of the TRIGIN input (J2). The sampling cycle is terminated and restarted by a rising edge on the TRIGIN line.



Description of Analog outputs



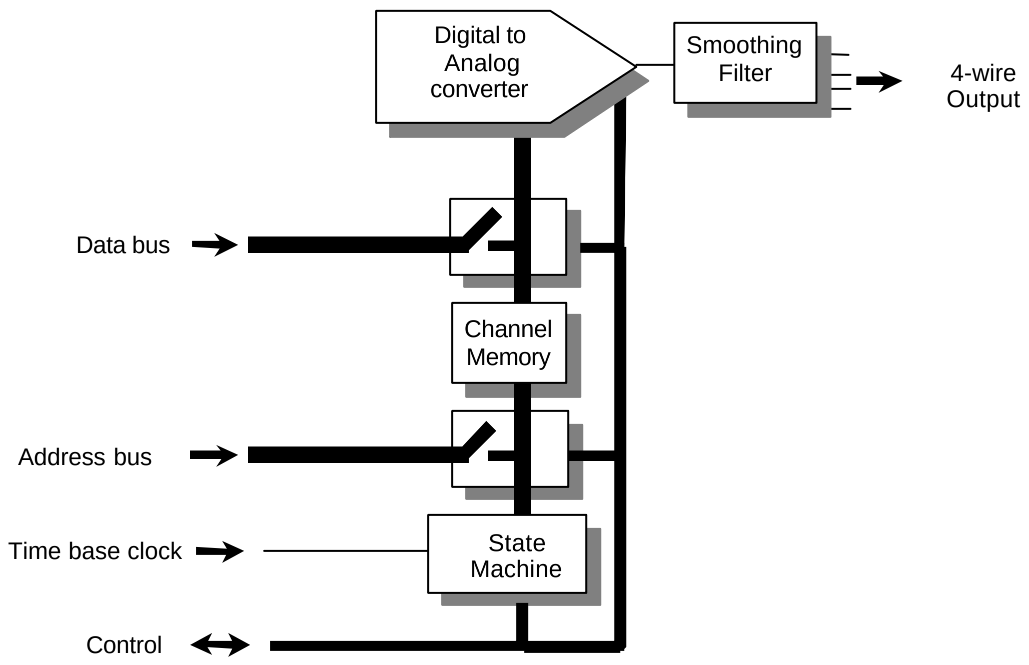
All the analog output channels are located on the second application card.
The analog output channels communicate with the VXI interface via an internal bus, which may be directly shared with the VXI bus (shared resource mode).
Each channel is independent and thus has available:

- a sampling time base
- a state machine
- an Digital to Analog converter
- a sampling memory
- a trigger signal multiplexer

All of these units are controlled by the decoder which manages the link between the different addressing areas listed above (pages 4-5 and 4-6) and the associated resources.

All the main functional blocks are strictly identical with those of the analog inputs, as the state machine, the converter and the sampling memory operate symmetrically with the analog inputs, as illustrated below:

State machine, converter and sampling memory



The state machine moves samples from channel memory to the digital/analog converter. The bus is available between two samples to exchange data in memory or to directly access the digital/analog converter.

5. Instrument Commands

Summary of SCPI Syntax

Conventions used in the definition

• Expressions contained in “ [ ] ” are optional.

Example **:** [SOURce]:VOLTage[:LEVel] [:IMMediate][:AMPLitude] means that this command may be written as:

“source:voltage:level:immediate”, “volt:lev”, “source:volt”, “volt”, "voltage:imm "etc.....

• The “**|**” symbol represents the "OR" function and indicates a choice.

Example : “<ON|OFF>” means that in this case both the ON and the OFF values are valid.

General rules of syntax

• Spaces (<a space>) may be inserted between a command and its parameters, or between parameters, but not within a command or a parameter (string type).

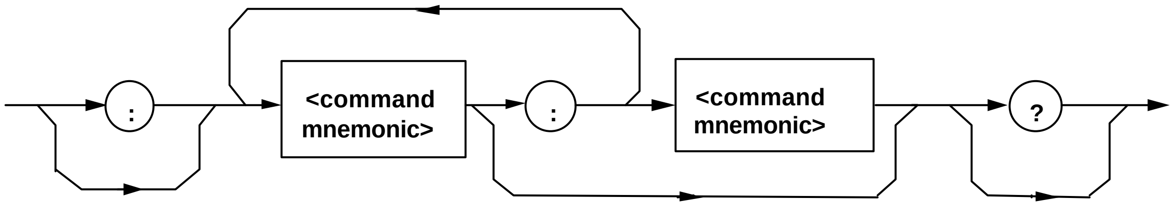
Example :
”volt 1,(@1)?” is incorrect, but “ volt   1,   (@1) “ is correct.

• Capital and lowercase letters may be used indifferently for command strings and parameters.

Example :
“VoLtaGe:Lev 3.5,(@5)" and “Form BIn” are correct.

Command format (Command Headers)

• An SCPI command (command header) is made up of key words (<command mnemonic>) separated by ":" and terminated by a question mark if a reply is expected (interrogative form). The command is separated from the first parameter (if it exists) by one or more spaces (or, to be precise, "<a space>") conforming to the IEEE 488.2 specification.



Example :
“output:state” or ” volt:level:imm?"

• A key word which makes up a command may be used in a condensed form (capital letters in the description) or in the long form (capitals and lowercase). The condensed form is made up of the first four letters of the long form. The last character must be a vowel only if the long form has four characters or less.

Example :
the key word SOURce may be written "source" (long form) or "sour" (condensed form), "LEVel" may be written "lev" or "level", "DATA" is invariable in both forms.

• If a key word is associated to a channel, it is still possible to use both forms.

Example :
the condensed form of "TTLTrig1" is TTLT1".

Parameters and separators

• A complete message (<command message>:command + associated parameters) is terminated either by an EOL character, a LF character (Line Feed: ASCII code 10, #H0A), or by a semi-colon (when many messages are written on the same line).

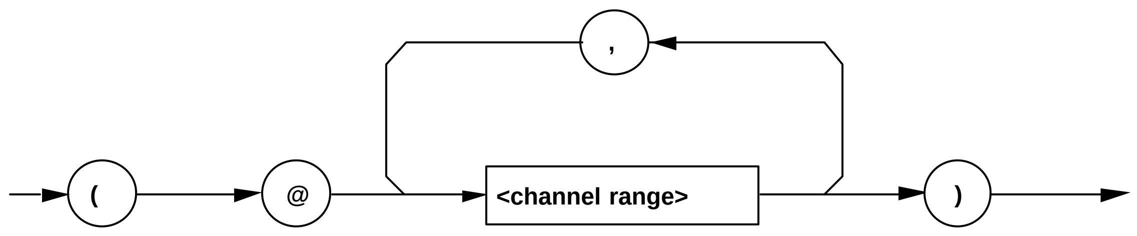
Example : “sour:volt 1,(@8);:outp:state? (@8) ”

• String parameters are governed by the same rules.

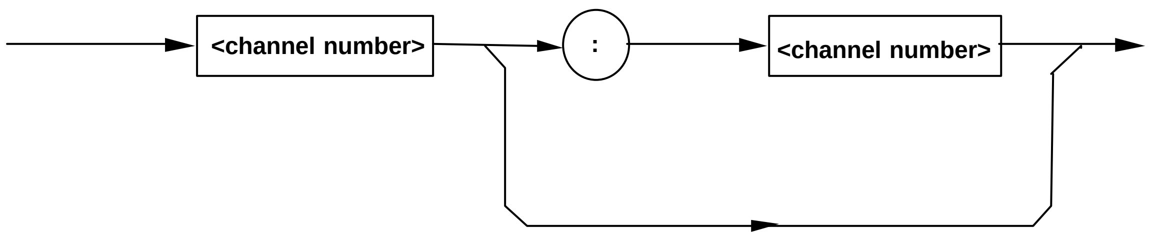
Example : the condensed form of the parameter "HEXadecimal" is "hex".

Channel list

• A “<channel_list>” is the set of channels concerned by a command. The syntax is as follows:



Where <channel



Examples:
"VOLT 2.5,(@5,7)" generates 2.5 V on channels 5 and 7.
"VOLT -9.98,(@6:8)" generates -9.98 V on channels 6,7 and 8.

Associated Commands

• When commands are included on the same line and separated by a ";", certain rules allow the description of simplified commands in a tree structure. The first command on the line implies a jump to the root of the tree, and following commands may omit the level already described by the preceding command.

Example :
"**trig:start:sour bus,(@1) ; immediate (@1)**" is equivalent to
trig:start:sour bus,(@1) ; trig:start:immediate(@1)

• The root may be accessed directly by placing a ":" before the first key word.

Input/Output Description

Key words

VOLTage: concerns analog input and output commands. The analog inputs and outputs are numbered from 1 to 8.

DATA: used when the value sent or read on the analog input/output channels is represented by an integer without units

LEVel: used when the value sent or read is the same as the voltage of the range currently in use.

SOURce: the SOURce keyword precedes all commands concerning analog outputs.

Examples:

“VOLT 1,(@5,6)” generates 1V on output channels 5 and 6 .
“SOURce:VOLTage:LEVel[:IMMediate][:AMPLitude] 0,(@8)” sets channel 8 to 0 volts.

Configuration

The program automatically recognizes the module's configuration (analog inputs or outputs) and adapts the error messages to correspond to this configuration.

This document is written for the standard configuration (6062A):
Four analog inputs: channels 1 to 4
Four analog outputs: channels 5 to 8

Range

Only one range may be used for the input/output channels: ± 10 V

Command Summary

SCPI Commands

Commands preceded by a "»" are SCPI 1994.0 confirmed commands.
The commands for the options are not included in this summary.

»**FORM**at				
»**SYST**em	:**PRES**et			
»		|:**VERS**ion?		
»		|:**ERR**or?		
		:**CLOC**k		
		:**VOLT**age	:**REF**erence	
»**STAT**us	:**OPER**ation	[:**EVEN**t]?		
»		|:**COND**ition?		
»		|:**ENAB**le		
		:**TRIG**ger	[:**EVEN**t]?	
			|:**COND**ition?	
			|:**ENAB**le	
»	:**PRES**et			
TEST?				
»[**SOUR**ce]	:**VOLT**age	[:**LEV**el]	[:**IMM**ediate]	[:**AMPL**itude]
		|:**DATA**		
»**MEAS**ure	:**VOLT**age	[:**LEV**el]	[:**IMM**ediate]	[:**AMPL**itude]?
		|:**DATA**		

TRIGger	:**TTLT**rg	[:**STAT**e]		
	|:**BUS**			
	|:**ETRG**			
	[:**IND**ependent]	:**STAR**t	[**IMM**ediate]	
	:**SYNC**hronous **|**			:**SOUR**ce
	[:**IND**ependent]	:**STOP**	**IMM**ediate]	
	:**SYNC**hronous **|**			**:VAL**ue
»	[:**SEQ**uence]	:**TIM**er		
»		:**COUN**t		
»**INIT**iate	[**IND**ependent]			
	|:**SYNC**hronous			
»**ABOR**t	[**IND**ependent]			
	|:**SYNC**hronous			
:**MODE**	:**SYNC**hronous			
»**MEM**ory	:**CLE**ar			
	:**LENG**th			
»	:**DATA**			
»**MMEM**ory	:**CLE**ar			
»	:**COPY**			
»	:**LOAD**			
»	:**STOR**e			
	:ADDed			
»**INP**ut	[:**STAT**e]			
»**OUTP**ut	[:**STAT**e]			
	:**TTLT**rig<n>	:**SOUR**ce		
		[:STATe]		
	:**SIGN**al	:**SOUR**ce		
	:**VINT**errupt			
FLASh	**:LOAD**	**:TRAC**e		

IEEE 488.2 Command

***RST**

***IDN?**

***OPT?**

***TST?**

***CLS**

***ESE**

***ESE?**

***ESR?**

***OPC**

***OPC?**

***SRE**

***SRE?**

***STB?**

***WAI**

***TRG**

Detailed command description

Data format and exchange commands

FORMat <character_string>

Description:
Defines the current format for the exchange of certain data types (integers). The format may be forced, irrespective of the standard format, by using the following prefixes: “#A” for the ASCii format, “#H” for HEXadecimal and “#B” for BINary. Every parameter concerned by this command and without a prefix will be considered as using this format. Every response concerned by this command will also use this format.

Parameters:
<character_string> may be “ASCii”, “HEXadecimal” or “BINary”.

ASCii : base 10, decimal.
HEXadecimal : base 16, hexadecimal
BINary : base 2, binary.

Initialization:
ASCii

Example:
“form hex” sets the default format to hexadecimal. Thus “volt 10,(@5)” will have the same effect as “volt #H10,(@5)” or “volt #A16”,(@5) and “volt? (@5)” returns the value “10”.

Interrogative form:
FORMat? ==> <character_string>

Note: In the rest of this document the standard FORMat used for each parameter will be preceded by a “*”.

Analog Channel Commands

Analog Output Channel Commands

OUTPut:[:STATe] <ON | OFF>, <channel_list>

Description:
Connects (ON) or disconnects (OFF) the specified channel on the user connector. After connection (ON), the output channel is set to 0v. This implies that the user must program a new value only when the specified channel is connected. If not, the programmed value will be lost when the channel is connected.

Initialization:
OFF

Example:
"OUTP ON,(@5:8)" connects all output channels.

Interrogative form:
OUTPut:[:STATe]? <channel_list> ==> <ON | OFF> {,<ON | OFF>}

****

[SOURce]:VOLTage[:LEVel][:IMMediate][:AMPLitude]
<numeric_value>,<channel_list>

Description:
Set the analog output voltage specified in the <channel_list>.

Parameter:
<numeric_value> gives the voltage expressed in VOLTS in the <NRf> format of the IEEE 488.2 standard. If the value exceeds the upper or lower limits, an error is returned.

Initialization:
0V on each analog output channel.

Example :
“VOLT 2.55,(@5,7:8)” sets the channels 5,7 and 8 at 2.55V.
“SOUR:VOLT:LEV:IMM:AMPL 10.0E-3,(@8)” sets channel 8 at 10 mV.

Interrogative form:
[SOURce]:VOLTage[:LEVel][:IMMediate][:AMPLitude]? <channel_list> ==> <numeric_value>

Response:
The output channel voltages in the order specified in the <channel_list>.

Example:
"VOLT? (@6,7) " ==> "2.5000,1.0000" Output voltages are 6: 2.5V, 7: 1V

****

[SOURce]:VOLTage:DATA[:IMMediate][AMPLitude]*<numeric_value>,<channel _list>

Description:
Sets the analog output channel voltages specified in the <channel_list>.
Unlike the LEVel commands, this command allows direct control of the digital/analog converter.

Parameters:
<numeric_value> is an integer value from 0 to #HFFFF (0 to 65535).

Conversion Table
<numeric_value>-->output voltage
#H0000 to #H7FFF0.0 $^{+}$V to 9.9997 V
#H8000 to #HFFFF-10.0 V to -0.0 V

Example:
"VOLT:DATA #H8000,(@7)" sets the output voltage on channel 7 to -10.0 V.
"FORM ASC ; VOLT:DATA 16384,(@7)" sets the output voltage on channel 7 to 5.0V.

Interrogative form:
[SOURce]:VOLTage:DATA[:IMMediate][AMPLitude]? <channel_list> ==> *<numeric_value>{, *<numeric_value>}

Response:
The voltage of the output channel in the order specified in the <channel_list>.

Analog Input Channel Commands

INPut:[:STATe] <ON | OFF>, <channel_list>

Description:
Connects (ON) or disconnects (OFF) the specified channel on the user connector.

Initialization:
OFF

Example:
"INP ON,(@1:4)" connects all input channels.

Interrogative form:
INPut[:STATe]? <channel_list> ==> <ON | OFF> {,<ON | OFF>}

****

MEASure:VOLTage[:LEVEL][:IMMediate][:AMPLitude]?<channel_list> ==> <numeric_value>{,<numeric_value>}

Description:
Immediately measures the input voltage of the channel chosen in the <channel_list>.

Response:
The voltage level expressed in volts, separated by commas in the order specified in the <channel_list>.

Example:
“MEAS:VOLT? (@1,3) ” ==> “7.5300,-2.9800” the voltage measured for channel 1 is 7.53V and -2.98V on channel 3.

Note: It is not possible to measure a channel when that channel is capturing data.

MEASure:VOLTage:DATA?<channel_list> ==> *<numeric_value>{,*<numeric_value>}

Description:
Identical to the preceding command, except that the value read is not converted into volts, but is presented as the raw data value (0 to #HFFFF) read at the output of the analog/digital converter.
The conversion table is the same as for the analog output.

Response:
Voltages are expressed in volts, separated by commas in the order specified in the <channel_list>.

Example:
“FORM HEX;:MEAS:VOLT:DATA? (@1,3) ” ==> “3FFF,8000” the voltage measured on input channel 1 is 5V and -10V on input channel 3.

Note: It is not possible to measure a channel when that channel is capturing data.

Trigger commands

General

TRIGger:TTLTrg[:STATe] <ON|OFF>

Description:
Enables (ON) or disables (OFF) VXI trigger lines (TTLTRIG) for use as trigger conditions.

Initialization:
ON

Interrogative form:
TRIGger:TTLTrg[:STATe]? ==> <ON | OFF>

****

TRIGger:BUS[:STATe] <ON|OFF>

Description:
Enables (ON) or disables (OFF) interruptions of type GET for use as trigger conditions.

Initialization:
ON

Interrogative form:
TRIGger:BUS[:STATe]? ==> <ON|OFF>

****

TRIGger:ETRG[:STATe] <ON|OFF>

Description:
Enables (ON) or disables (OFF) interrupts from External Trigger lines for use as trigger conditions.

Initialization:
ON

Interrogative form:
TRIGger:ETRG[:STATe]? ==> <ON|OFF>

****

TRIGger[:SEQuence]:TIMer <numeric_value>, <channel_list>

Description:
Set the sampling period for the channels in the <channel_list>.

Parameters:
<numeric_value> is an integer value; the time unit is the microsecond; the minimum accepted value is 10µs for analog inputs and 2µs for analog outputs; the maximum accepted value is 1000000 µs (1 second).
If an external reference clock is used (see the SCPI “SYStem:CLOCk” command) the period that is programmed is:
(<num_value> * 10/Freq_{ext}) µsec

Example: if the user wishes to program 100 µsec with Freq_{ext} = 16MHz then <num_value> = 160

Initialization:
1,000000 indicate 1 second.

Example:

Reference clock of the **internal** timer (10MHz) :
"TRIG:TIM 10,(@1)" , the sampling frequency on channel 1 is 100KHz.
"TRIG:TIM 1000,(@5:8)" , the sampling frequency on channels 5, 6, 7, 8 is 1KHz.

Reference clock of the **external** timer with Freq_{ext} = 16 MHz :
"TRIG:TIM 160,(@1)" , the sampling frequency on channel 1 is 10KHz
"TRIG:TIM 1000,(@5:8)" , the sampling frequency on channels 5, 6, 7, 8 is 1600Hz.

Interrogative form:
TRIGger:[SEQuence]:TIMer ? <channel_list> ==> <numeric_value>

****

TRIGger:[SEQuence]COUNt <numeric_value>, <channel_list>

Description:
Sets the number of times that the trace associated with the specified channel is scanned.

Parameters:
<numeric_value> is an integer value from 0 to 65535

Note *A value of zero starts an infinite loop*. The cycle continues until an ABORt or a STOP command is received.

Initialization:
0, an infinite loop.

Interrogative form:
TRIGger:[SEQuence]COUNt? <channel_list> ==> <numeric_value>

MODE:SYNChronous <ON|OFF>,<port_num>

Description:
Selects the operating mode for the selected application port: either synchronous mode (ON) or independent mode (OFF).

Parameter:
<port_num>:either the value "1" or "2", corresponding to the number of the application port for which the operating mode is to be set.
The 6062 module includes two application ports as follows:

- 6062A: port number 1 = 4 inputs (channels 1 to 4)
  port number 2 = 4 outputs (channels 5 to 8)

- 6062B4: port number 1 = 4 outputs (channels 1 to 4)

- 6062B8: port number 1 = 4 outputs (channels 1 to 4)
  port number 2 = 4 outputs (channels 5 to 8)

- 6062C4: port number 1 = 4 inputs (channels 1 to 4)

- 6062C8: port number 1 = 4 inputs (channels 1 to 4)
  port number 2 = 4 inputs (channels 5 to 8)

Port 1 corresponds to connectors J1 and J2 on the front panel, while port 2 corresponds to connectors J6 and J7.

Initialization:
OFF (independent mode)

Examples:
“MODE:SYNC ON,1” sets application port 1 to synchronous mode
“MODE:SYNC OFF,2” sets application port 2 to independent mode.

Interrogative form:
MODE:SYNChronous? <port_num> ==> <ON|OFF>

Independent mode

Each channel may be started or stopped independently of the other channels.

START Commands

TRIGger[:INDependent]:STARt:SOURce <character_string>, <channel_list>

Description:
Sets the trigger source so that exchange between the input/output channels and the associated trace memories may begin.

Parameters:
<character_string> accepts the following values:

IMMediate : No waiting for a particular event. As soon as the INITitate command is received, exchange between the input/output channels and the associated trace memories begins at the sampling frequency.

HOLD : Inhibits the trigger source. Only the "TRIGger[:INDependent]:STARt[:IMMediate]" command can start an exchange.

TTLTrg0, TTLTrg1, TTLTrg2, TTLTrg3, TTLTrg4, TTLTrg5, TTLTrg6, TTLTrg7: The trigger event to be used is the activation of the TTLTRIG line specified, if it was previously enabled by the "TRIGger:TTLTrg[:STATe] ON" command.

BUS : The trigger event to be used is the reception of a trigger by the word serial protocol (Word Serial Trigger), or the IEEE 488.2 "*TRG" command, if it was previously enabled by the "TRIGger:BUS[:STATe] ON" command.

ETRG: The trigger event to be used is the activation of the External Trigger associated to the specified channel, if it was previously enabled by the "TRIGger:ETRG[:STATe] ON" command.

Initialization:
IMMediate

Example:
"TRIG:STAR:SOUR TTLT1,(@1)" , sets the trigger for channel 1 to the TTLTRIG1 line.

Interrogative form:
TRIGger[:INDependent]:STARt:SOURce? <channel_list> ==> <character_string>

****

TRIGger[:INDependent]:STARt[:IMMediate] <channel_list>

Description:
Although the SOURce trigger has already been selected (START trigger), the trigger will take effect as soon as this command is received. The trigger must first be initialized by the "INITiate[:INDependent] <channel_list>" command; if not, this command is ignored.
This command is normally used when the SOURce trigger is on HOLD.
This command is one of the software triggers.

Examples:
"TRIG:SOUR HOLD (@1)" sets the trigger source for channel 1 to the HOLD state.
"INIT (@1)" Initializes the trigger cycle.
"TRIG:STARt (@1)" starts exchanges between channel 1 and the exchange memory.

No interrogative form

****

STOP Commands

TRIGger[:INDependent]:STOP:VALue <ON|OFF>, <numeric_value>, <channel_list>

Description:

Allow to choose a voltage to apply on the output upon a hard STOP trigger reception (ETRG or TTLTrig).

Parameters:

<ON|OFF> : ON allows this mode, OFF disables it.

<numeric_value> gives the voltage expressed in VOLTS in the <NRf> format of the IEEE 488.2 standard. If the value exceeds the upper or lower limits, an error is returned.

Initialization:

OFF

<numeric_value> = 0V.

No interrogative form.

****

TRIGger[:INDependent]:STOP[:IMMediate] <channel_list>

Description:

Although SOURce was selected as the STOP event, the trace command is effectively stopped as soon as this command is received. This command is normally used when the SOURce is on HOLD.

This is a software command.

No interrogative form

INITiate and ABORT Commands

INITiate[:INDependent] <channel_list>

Description:
This command initializes all the specified trace's memory pointers associated with the input/output channels to the beginning of the trace. It validates the TRIGger STARt conditions.
The trigger cycle concerns only the channels specified in the <channel_list>; other channels are not affected by this command.

No interrogative form

****

ABORt[:INDependent] <channel_list>

Description:
This commands aborts the entire cycle for the specified channels. A new INITiate command is required to re-start the cycle.
After this command, the associated analog outputs are reset to zero.

No interrogative form

Synchronous mode

The four selected application port channels are started or stopped at the same time.

STARt Commands

TRIGger:SYNChronous:STARt:SOURce <character_string>, <port_num>

Description:
Sets the trigger source so that exchanges between the input/output channels and the associated trace memories may begin.

Parameters:
<character_string> accepts the following values :

IMMediate : No waiting for a particular event. As soon as the INITiate command is received, exchanges between the input/out channels and their trace memories begin at the specified sampling frequency.

HOLD : Inhibits all trigger sources. Only the "TRIGger:SYNChronous:STARt[:IMMediate]" command can activate exchanges.

TTLTrg0, TTLTrg1, TTLTrg2, TTLTrg3, TTLTrg4, TTLTrg5, TTLTrg6, TTLTrg7: The trigger event used is the activation of the specified TTLTRIG line, if it was previously enabled by the "TRIGger:TTLTrg[:STATe] ON" command.

BUS : The trigger event used is the reception of a trigger by the word serial protocol (Word Serial Trigger), or the IEEE 488.2 "*TRG" command, if it was previously enabled by the "TRIGger:BUS[:STATe] ON" command.

ETRG: The trigger event used is the activation of the External Trigger associated to the channel specified, if it was previously enabled by the "TRIGger:ETRG[:STATe] ON" command

<port_num> : see the "MODE:SYNChronous" command.

Initialization:
IMMediate

Example:
"TRIG:SYNC:STAR:SOUR TTLT6,1" , the trigger to be used for the four channels on port 1 is the TTLTRIG6 line.

Interrogative form:
TRIGger:SYNChronous:STARt:SOURce?<port_num>==> <character_string>

****

TRIGger:SYNChronous:STARt[:IMMediate] <port_num>

Description:
Although the SOURce trigger has already been selected (START trigger), the trigger will take effect as soon as this command is received. Of course, the trigger must first be initialized by the "INITiate[:SYNChronous <port_num>" command; if not, this command is ignored. This command is normally used when the SOURce trigger is on HOLD. This command is one of the software triggers.

Examples:
“TRIG:SYNC:STAR:SOUR HOLD 2” sets the trigger source for the four channels on port 2 (corresponding to channels 5, 6, 7 and 8) to the 'HOLD' state.
“INIT:SYNC 2” Initializes the trigger cycle for port 2.
“TRIG:SYNC:STARt 2” starts the exchanges between the four channels and their associated exchange memories.

No interrogative form

****

STOP Commands

TRIGgerSYNChronous:STOP:VALue <ON|OFF>, <numeric_value>, <port_num>

Description:
Allow to choose a voltage to apply on the four output of the port upon a hard STOP trigger reception (ETRG or TTLTrig).

Parameters:
<ON|OFF> : ON allows this mode, OFF disables it.

<numeric_value> gives the voltage expressed in VOLTS in the <NRf> format of the IEEE 488.2 standard. If the value exceeds the upper or lower limits, an error is returned.

Initialization:
OFF
<numeric_value> = 0V.

No interrogative form.

****

TRIGger:SYNChronous:STOP[:IMMediate] <port_num>

Description:
Although a STOP event was selected as SOURce, the exchanges will not actually halt until this command is received. This command is normally used when the SOURce is in the HOLD state.
This is a software command.

No interrogative form

INITiate and ABORT Commands

INITiate:SYNChronous <port_num>

Description:
This command initializes all trace memory pointers for the port specified at the beginning of the trace. It validates the TRIGger STARt conditions.
Thereafter, as soon as a TRIGger STARt is received, the four channels of the port will begin their capture or generation cycle at the same time.

No interrogative form

****

ABORt:SYNChronous <port_num>

Description:

This command aborts the entire cycle for the four channels of the specified port. A new INITiate command is necessary to re-start the cycle.
After this command, the port's analog outputs are set to zero.

No interrogative form

Memory Management

Channel Memory Manager

The memory manager controls the different trace memories associated with the input/output channels. The available memory size per channel is 65534 samples.

MEMory:LENGth <numeric_value>, <channel_list>

Description
Define the trace depth, limited to 65534 samples allocated per channel. The trace may contain either samples from a capture cycle on an input channel, or samples loaded by the user in order to transfer them to the output in question.
The generation or capture of samples on the channels specified in the <channel_list> uses the depth specified in (LENGth).

Initialization :
65534

Parameter:
<numeric_value> is an integer value from 1 to 65534 .

Example:
"MEM:LENG 100,(@1:4)" , channels 1,2,3 and 4 reserve 100 samples for exchanges between the trace memory and the input/output channels.

Interrogative form:
MEMory:LENGth? <channel_list> ==> <numeric_value>, returns the depth defined for each channel specified in the <channel_list>.

****

MEMory:CLEar <channel_list>

Description:
This command erases all the data stored in the channel memory.

Note *This command only concerns the channel memories used in generation (analog outputs).*

Examples:
"MEM:CLE (@5:8)" , erases all the data contained in the trace memory for channels 5,6,7 and 8 (4 * 65534 samples erased) .

No Interrogative form

****

MEMory:DATA <channel_num>, *<data>{,*<data>}

Description:
This command fills the channel memory for the selected channel with the specified <data>. The data is transmitted in comma delimited ASCII format.

Note *This command only concerns output channels.*

This SCPI command is particularly useful if the number of points to be loaded is not too large, since it is relatively slow compared to other methods. It is often advantageous, although less flexible, to utilize direct VXI bus transfers towards the trace memories.
To load data, the depth of the specified trace memory must be defined by the SCPI "MEMory:LENGth ...".command.

Note *The user may change one or many data points at any time, even if a generation cycle is under way.*

Parameters:
<channel_num>: the channel number on which the operation will take place.

<data> : an integer value from 0 to 65535 (0 to #HFFFF) transferred in an ASCII data list.

Examples
"MEM:LENG 2,(@5)"
"MEM:DATA 5,32768,16384" , fills the first two samples in the channel memory for channel 5 with corresponding values to -10V and +5V.
"MEM:LENG 4,(@6)"
"FORM HEX"
"MEM:DATA 6,8000,C000,3FFF,0000" , fills the four samples of the memory for channel 6 with values corresponding to -10V, -5V, 5V and 0V..

Interrogative form:

MEMory:DATA? <channel_num>[,<format>] ==><*data>{,*<data>}

Description:
This command retrieves data in the reserved channel memory. Data may be retrieved as a comma delimited ASCII format.

Note *This command only concerns input channels.*

This SCPI command is particularly useful if the number of points to be loaded is not too large, since it is relatively slow compared to other methods. To recover data, the specified trace memory must be defined by the SCPI "MEMory:LENGth ..." command.

Parameters:
<channel_num>: channel number on which the transfer will take place.

<format> : the expected format of the data.
If this parameter is given, it takes precedence over the currently specified "FORMat".
"ASCii", "HEXadecimal", "BINary" = identical to the FORMat parameter.

Examples:
"MEM:LENG 3,(@1) ; FORM HEX"
"MEM:DATA? 1", retrieves the three samples from the trace memory of channel 1 ==> #H07FF ,#H0020, #H1FF.
"MEM:DATA?1,ASC" ==>4095,32,511

Mass storage memory manager

The mass storage memory manager controls the mass storage of samples for later use. Its memory size is 524288 samples (1 Mbytes).

Note The relative commands to the mass storage memory are allow if the MEMory option is present.

MMEMory:CLEar [<index>[,<depth>]]

Description:
This command erases the data stored in mass memory from the specified index

Optional Parameters: The mass memory will be erased from this index.
Default : 1

<depth> : an integer value between 1 and 524288 (512*1024)
Default : 524288 - (<index> - 1)

Examples:
"MMEM:CLE " , erases all data from the beginning of mass storage memory.
"MMEM:CLE 10" , erases all the data contained in mass storage memory from the tenth block on.
"MMEM:CLE 1,10" , erases the first ten samples stored in mass memory.
"MMEM:CLE 524000,2050" , generates an "out of memory" error.

No Interrogative form

****

MMEMory:COPY <source_index>,<destination_index>,<depth>

Description
This command copies a block of data from one part of mass memory to another. This data is transferred by internal DMA (Direct Memory Access).

Parameters:
<source_index> : an integer value between 1 and 524288.

<destination_index> : an integer value between 1 and 524288.

The value of <source_index> must be greater than the value of <destination_index>.

<depth> : an integer value between 1 and 524288.

Examples:
"MMEM:COPY 20,1,10" , copies ten samples from the twentieth at the beginning of the mass memory.
"MMEM:COPY 1,262144,262144" , copies the first half of the mass memory to the second one.
"MMEM:COPY 1,512,2000" , generates an "out of memory" error.
"MMEM:COPY 1,10,20" , generates an "out of memory" error.

No Interrogative form

****

MMEMory:DATA <index>,<depth>, *<data>{,*<data>}

Description:
This command fills the mass memory from an index with the specified <data>. The data is transmitted in comma delimited ASCII format.
User can load block of data anywhere in the mass memory. After fill it, he can load these blocks in channel memory using the MMEM:LOAD command described later.
The number of data passed is fixed by the <depth> parameter.

Parameters:
< index> : an integer value between 1 and 524288.
<depth> : an integer value between 1 and 524288.
<data> : an integer value from 0 to 65535 (0 to #HFFFF) transferred in an ASCII data list.
For information, the VXI input buffer is fixed at 10,000 bytes. So, a maximum of data can be passed by this command . It depends of the selected format (about 1995 values for an Hexadecimal format).

Examples
"FORM HEX"
"MMEM:DATA 1,4,8000,3FFF,0000,7FFF" , fills the first four samples of the mass memory with corresponding values to -10V, +5V, 0V and +10V.
"MMEM:LOAD 1,1,(@5)" transfer the four data defined above in the first part of the channel memory 5.
"MMEM:DATA 10000,2,7FFF,0000" , fills two samples of the mass memory located at index 10,000 with values corresponding to 5V and 0V.
"MMEM:LOAD 10000,5,(@5)" , concatenate the two new data after the first defined block in channel memory 5.

MMEMory:LOAD <source_index>,<destination_index>[,<depth>],<channel_list>

description:
Transfer data between mass storage memory and channel memory. The data is transferred by internal DMA.

Note *This command only concerns output channels.*

Parameters:
<source_index> : an integer value between 1 and 524288, and which is part of mass storage memory.

<destination_index> : an integer value between 1 and 65534, and which is part of the trace memory associated with the channels specified in the <channel_list>.

<depth> : an integer value between 1 and 65534.
If this parameter is omitted, its default value is the predefined depth for the channel memory (defined with "MEMory:LENGth").

Examples:
"MEM:LENG 65534,(@5:8)
"MMEM:LOAD 1,1,(@5:8)" ; loads the 65534 samples from the channel memory for output channels 5,6,7 and 8 with the 65534 first samples of mass storage memory.
"MMEM:LOAD 1,100,10,(@5)" , loads the first ten samples from the hundredth sample of the mass storage memory into the channel memory 5.

No Interrogative form

MMEMory:STORe <source_index>,<destination_index>[,<depth>], <channel_list>

Description:
Transfers data between the channel memory associated with one or many channels and mass storage memory. Internal DMA is used for this transfer.

Note *This command only concerns input channels.*

Parameters:
<source_index> : an integer value from 1 to 65534, which is part of the channel memory associated with the channels specified in the <channel_list>.

<destination_index> : an integer value between 1 and 524288, which is part of mass storage memory.

<depth> : an integer value between 1 and 65534.
If this parameter is omitted, its default value is the predefined depth for the channel memory (defined with “MEMory:LENGth”).

Examples:
“MEM:LENG 1000,(@1)”
"MMEM:STOR 1,1,(@1)" ; fills the first 1000 samples in mass storage memory with the first 1000 samples of the trace memory for channel 1.
"MMEM:STOR 1,1,10,(@3)", loads the first ten samples of the trace memory for channel 3 into the first ten samples of mass storage memory.

No Interrogative form

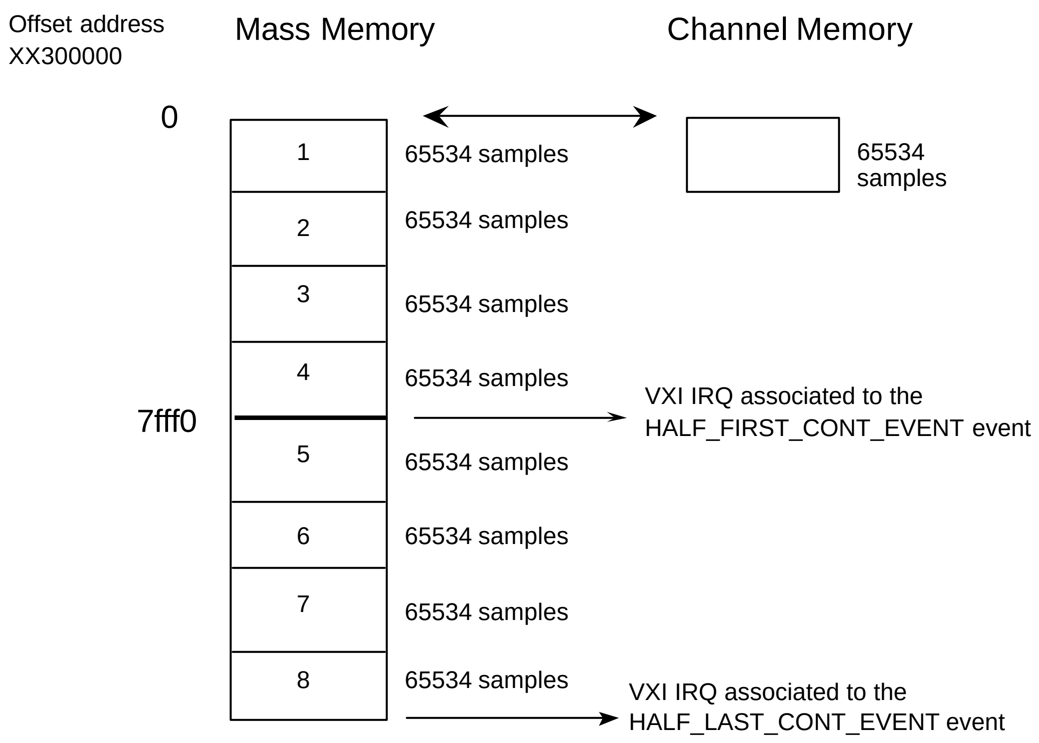
MMEMory:ADDed <data_string>[,<num>],

Description:

Automatically allows to increase the size of the channel memory by about 512k samples by using the mass storage memory .

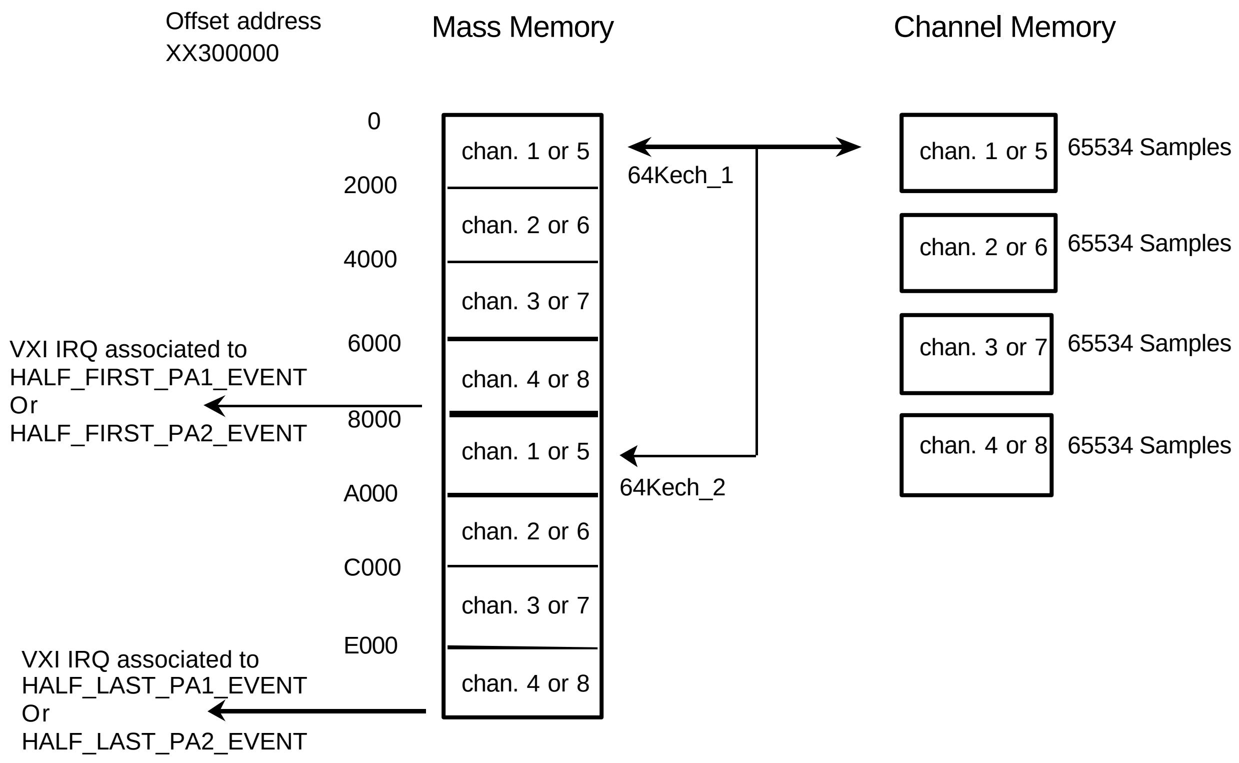
Parameters:
<data_string> can take values:

ONE:
One of the eight channel memories selected by the optional parameter **<num>** (value between 1 and 8), will have a size of 65534 Samples + 524272 (about 576k samples).
The module takes on transfers between the channel memory and the mass storage memory in DMA mode.
A VXI interrupt associated with the HALF_FIRST_CONT_EVENT event (value : 40), is generated to inform the user that the first half of the mass storage memory is full.
A VXI interrupt associated with the HALF_LAST_CONT_EVENT event (value : 45), is generated to inform the user that the second half of the mass storage memory is full.
So, the user will can fill or acquire the 1st half of the mass storage memory, in shared memory mode, by a block of 4*65534 samples (from address XX300000), on the HALF_FIRST_CONT_EVENT event reception, as well as the second half of the mass storage memory (from address XX37FFF0) on the HALF_LAST_CONT_EVENT event reception. In this way, the channel will can generate or acquire a signal in continuous mode on a depth of about 576k samples.



FOUR:
The 4 channels of the selected port with the optional parameter **<num>** (value between 1 and 2) will see their memory size doubled to pass from 65534 samples to 131068 samples.
The module takes on transfers between the channel memory and the mass storage memory in DMA mode.
A VXI interrupt associated with the HALF_FIRST_PA1_EVENT event (value : 32) or the HALF_FIRST_PA2_EVENT event (value : 34) depending of the used port, is generated to inform the user that the first half of the mass storage memory is full.
A VXI interrupt associated with the HALF_LAST_PA1_EVENT event (value : 33) or the HALF_LAST_PA2_EVENT event (value : 35) depending of the used port, is generated to inform the user that the first half of the mass storage memory is full.
The user will can, on one of the two events reception, fill or acquire samples by block by using the shared memory mode, with the respect of the added memory address describe on the next figure.

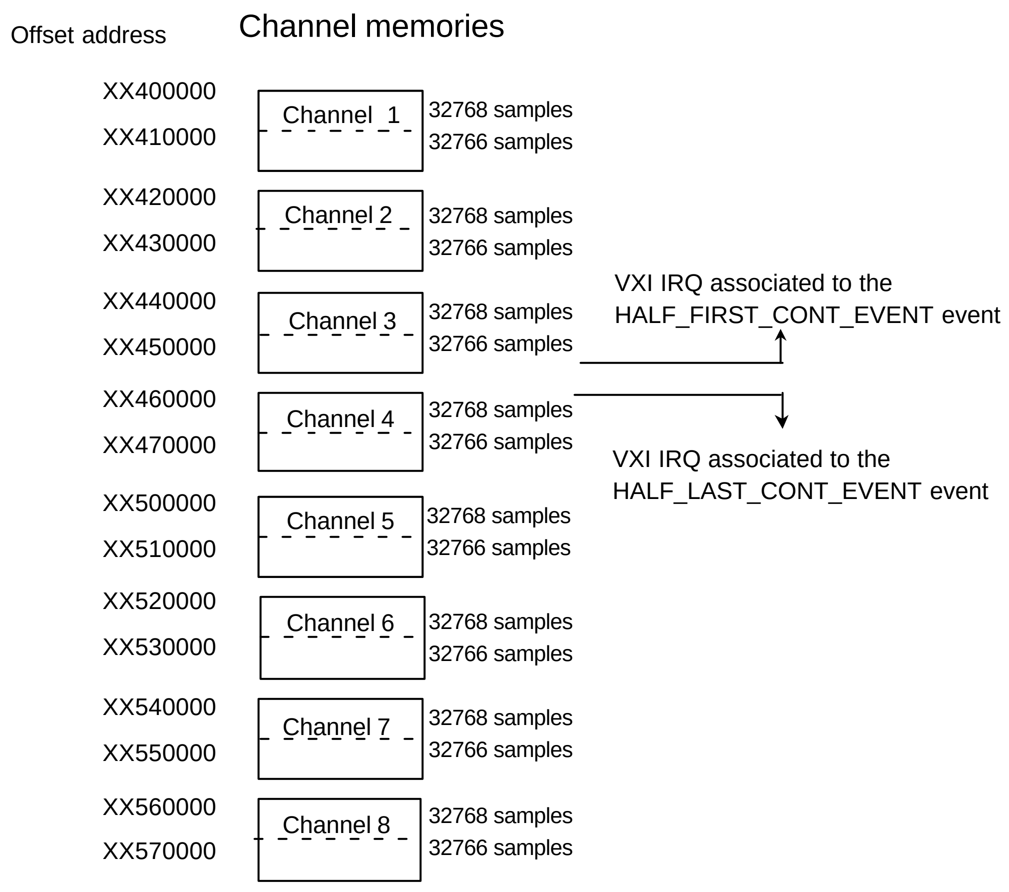
Note The user will check that the 4 channels of the concerning port begin their acquisition or generation cycle in the same time. To do this, chose a TTLTrig or ETRG trigger (by connecting together the 4 inputs TRIGIN of the front panel connector J2 or J7) in independent mode or pass in synchronous mode.



EIGTh:
It is a different mode where the mass storage memory is not used and the affected size to the channel memories remains of 65534 samples.
This mode is only used to generate or to acquire in continuous mode on all the channels.
The module informs the user by generate a VXI interrupt associated with the HALF_FIRST_CONT_EVENT event (value : 40) when the 8 channel memories have reach the half of their memory size (32768 samples) and a VXI interrupt associated with the HALF_LAST_CONT_EVENT event (value : 45) when the eight channel memories have reach their last sample.
On the HALF_FIRST_CONT_EVENT event reception, the user should fill or acquire the eight channel memory blocks corresponding to the first 32768 samples and on the HALF_LAST_CONT_EVENT event reception, the user should fill or acquire the eight channel memory blocks corresponding to the last 32766 samples. See the next figure.

The optional parameter <**num**> is not gives.

Note The user will check that the 4 channels of the concerning port begin their acquisition or generation cycle in the same time. To do this, chose a TTLTrig or ETRG trigger (by connecting together the 4 inputs TRIGIN of the front panel connector J2 or J7) in independent mode or pass in synchronous mode.



NONE:
Allow to inhibit this "added memory" mode.

Initialization:
NONE

No interrogative form.

Commands related to OUTPut

OUTPut:SIGNal:SOURce <data_string>,<channel_list>

Description:
Select the signal, which will be sent to the TRIGOUT pin (corresponding to the channel selected in the <channel_list>) of the trigger connectors (J2 or J7) on the front panel.

Parameter:
<data_string>: may contain one of the following values:

RUN_stop:
This signal is at logical 1 when the channel is in operation (data capture or generation cycle) and at logical 0 when the channel is stopped.

TIMer:
This signal represents the programmed sampling period corresponding to the channel.

HALF:
When the channel is in operation (data capture or generation cycle), this signal goes to 1 each time the pointer on the channel memory reaches 32k samples (half of the maximum depth), and goes back to 0 when the pointer points to the first sample.

CYCLe:
When the channel is in operation (data capture or generation cycle), this signal changes state each time the pointer on the channel memory has executed a cycle.

Initialization:
RUN_stop

Example:
"OUTP:SIGN:SOUR TIM,(@5)", visualizes the sampling frequency for channel 5 on the TRIGOUT1 pin of the J7 connector. Of course, an INITiate command must have been sent first.

Interrogative form:
"OUTPut:SIGNal:SOURce? <channel_list>" ==> <data_string>

OUTPut:TTLTrig<n>:SOURce <data_string>,<channel_list>

Description:
Allow to affect one of the eight TTLTrig lines to the specified trigger source relative to one or several channels. It just corresponds to connect a TTLTrig line onto one of the TRIGOUT pins of the J2 or J7 connector (see the command here above "OUTP:SIGNal:SOURCe")

Parameters:
<n> : can take values from 0 to 7 and corresponding to the selected TTLTrig line number.

<data_string>: can take the following values:

RUN_stop:
Connect the TTLTrig<n> line onto the RUN_stop signal of the selected channel in <channel_list>. This signal is at logical 1 when the channel is in operation (data capture or generation cycle) and at logical 0 when the channel is stopped, it means that the TTLTrig line is active when the channel is stopped.

TIMer:
Connect the TTLTrig<n> line onto the TIMer signal of the selected channel in <channel_list>. This signal represents the programmed sampling period corresponding to the channel.

HALF:
Connect the TTLTrig<n> line onto the HALF signal of the selected channel in <channel_list>. When the channel is in operation (data capture or generation cycle), this signal goes to 1 each time the pointer on the channel memory reaches 32k samples (half of the maximum depth), and goes back to 0 when the pointer points to the first sample.

CYCLe:
Connect the TTLTrig<n> line onto the HALF signal of the selected channel in <channel_list>. When the channel is in operation (data capture or generation cycle), this signal changes state each time the pointer on the memory channel has executed a cycle.

Initialization:
NONE ==> no connected signal.

Example:
"OUTP:TTLT7:SOUR CYCLE,(@5)", allows to connect TTLTrig7 line onto the CYCLe signal of channel 5.

Interrogative form:

"OUTPut:TTLTrig<n>:SOURce? <channel_list>" ==><data_string>

****

OUTPut:TTLTrig<n>[:STATe] <ON|OFF>,<channel_list>

Description:
Allows (ON) or forbids (OFF), the selected choice at the time of the here above command "OUTP:TTLTrig:SOURce".

Parameter:
<n> : can take values from 0 to 7 and corresponding to the selected TTLTrig line number.

Initialization:
OFF

Interrogative: form
"OUTPut:TTLTrig<n>[:STATe]? <channel_list>" ==><ON|OFF>

OUTPut:VINTerrupt[STATe] <ON|OFF>,<channel_list>

Description:
Allows (ON) or forbids (OFF) the emission of a VXI interrupt associated to a particular event relative to the selected channels in <channel_list>.
A interrupt can be generated at any level changing of one of TRIGOUT pins of J2 or J7 connector affected by the SCPI commands "OUTPut:SIGNal:SOURce" or "OUTPut:TTLTrig:SOURce"

Value list of VXI events (see the VXI bus 1.4 section E.4 "Protocol Event" specification") writes in the Status/Id register while an interrupt acknowledge cycle:

RUN_HIGH_EVENT	0 + channel number -1 (from 0 to 7)
RUN_LOW_EVENT	8 + channel number -1
TIM_HIGH_EVENT	16 + channel number -1
TIM_LOW_EVENT	24 + channel number -1
HALF_HIGH_EVENT	32 + channel number -1
HALF_LOW_EVENT	40 + channel number -1
CYCLE_HIGH_EVENT	48 + channel number -1
CYCLE_LOW_EVENT	56 + channel number -1

The 'HIGH' type events are generated on a rising edge of the TRIGOUT pin and those of 'LOW' type are generated on a falling edge of this same pin.

The events listed below are generated even if the parameter of the "OUTPut:VINTerrupt" command equal "OFF" and they correspond to the "added memory" mode. (see the "MMEMory:ADDed" command).

HALF_FIRST_CONT_EVENT	40
HALF_LAST_CONT_EVENT	45
HALF_FIRST_PA1_EVENT	32
HALF_LAST_PA1_EVENT	33
HALF_FIRST_PA2_EVENT	34
HALF_LAST_PA2_EVENT	35

Warning: No interruption will be generated if the TRIGOUT channel pin is connected onto the sampling frequency ("OUTP:SIGN:SOUR TIM,<channel_list>") and that this one is lower that 100 µsec

Initialization :
OFF

Examples:

```
"OUTP:SIGNAL:SOUR TIMer,(@5)"
"OUTP:VINT ON,(@5)"
"INIT (@5)"
```

==> at any rising edge of the clock used to the channel 5 sampling frequency, a VXI interrupt associated to the "20" event is generated and at any falling edge, a VXI interrupt associated to the "28" event is generated.

```
"OUTP:SIGNAL:SOUR CYCLE,(@1)"
"OUTP:VINT ON,(@1)"
"INIT (@1)"
```

==> at any acquisition cycle of channel 1, a VXI interrupt will be generated alternatively to the "48" and "56" event.

Interrogative form:

```
OUTPut:VINTerrupt[:STATe]? <channel_list>  ==> <ON|OFF>
```

Special Commands

FLASh:LOAD:TRACe <source_signal>,<num_signal>,<channel_list>

Description:
Allow to transfer predefined waveforms (stored in Flash memory) to the channel memories.

Parameters:
<source_signal>: type of waveform to transfer. This parameter can take values:

SINE: sine signal, 50 samples by period.
SAWtooth: sawtooth signal, 50 samples by period.
SQUare: square signal, 20 samples by period.
TRIangle: triangle signal, 50 samples by period.
USER: "user" signal created by the "FLASh:SIGNal<n>" command.

<num_signal>: corresponds to six different amplitudes for all waveforms excepted for the "USER" signals. This parameter can take the following values:

1: 0 to 1V.
2: -1V to +1V.
3: 0 to 5V.
4: -5V to +5V.
5: 0 to 10V.
6: -10V to +10V.

For the "**USER**" signal , <**num_signa**l> corresponds to one of the two user signals defined by the "FLAS:SIGN<n>" command.
1: signal n°1 defined by "FLAS:SIGN1 ..."
2: signal n°2 defined by "FLAS:SIGN2 ..."

Examples:
"FLAS:LOAD:TRAC SINE,6,(@5,6)"
"FLAS:LOAD:TRAC SQUare,1,(@7,8)"
"TRIG:TIM 2,(@5)" sampling period of channel 5: 2µsec
"TRIG:TIM 10,(@6)" sampling period of channel 6: 10µsec
"TRIG:TIM 100,(@7)" sampling period of channel 7: 100µsec
"TRIG:TIM 1000,(@8)" sampling period of channel 8: 1000µsec
"OUTP ON,(@5:8)" connects the output relays
"INIT (@5:8)" start the generation on the four outputs.
channel 5 : sine 10kHz ,+-10V
channel 6: sine 2kHz, +-10V
channel 7: square 500Hz, 0-1V
channel 8: square 50Hz, 0-1V

No interrogative form

FLASh:SIGNal<n'> <Step>,<A0>,<A1>,<B1>,...,<An>,<Bn>

Description:
Allow to define a waveform factorized in Fourier series, which will be stored in Flash memory (not volatile).

$$S(t) = A0 + \sum_{n=1}^{n=10} \left[An.Cos\left(\frac{n.\omega.t}{Step} \right) + Bn.Sin\left(\frac{n.\omega.t}{Step} \right) \right]$$

Parameters:

n' : Two signals can be memorized. **n'** corresponds to the signal number to define and it can take value 1 or 2.

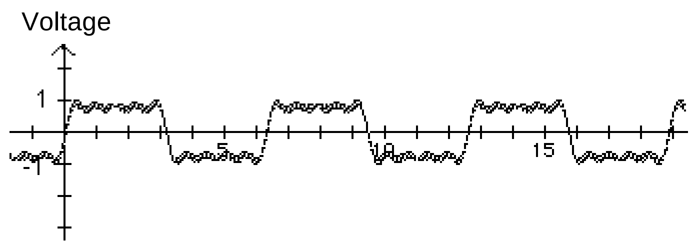
Step : defines the sample number by period. If Step = 100 and the sampling period equal 10 then the signal period will be 1 millisecond.

A0 : signal offset of value between -10V to +10V.

A1,B1,...,An,Bn: coefficients of the Co-sine and Sine terms of value between -10V and -10V with n=10 (so 20 parameters to enter) .

Examples:

Factorization in Fourier series of a square signal until row 9:
y = sin x + 1/3 (sin 3x) + 1/5 (sin 5x) + 1/7 (sin 7x) + 1/9 (sin 9x)



To define this signal in 100 points and to memorize it in Flash memory, there is only need to send the command:
"FLAS:SIGN1 100,0,0,1,0,0,0,0.333,0,0,0,0.2,0,0,0,0.142,0,0,0,0.111,0,0"

To define a Cosine signal with an offset of -5V: y = -5 + COs x
"FLAS:SIGN2 100,-5,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0"

Voltage

No interrogative form.

Auto-test

TEST? <channel_list> ==> <data_string>

Description:
Executes the self-test on the specified channel and returns the test result.

Response:
"PASSED" if the test reveals no errors.
"FAILED" if the test reveals errors.

Example:
“TEST? (@1:4)” ==> “PASSED,PASSED,FAILED,PASSED”, channel 3 has an error, while the others are OK.

Note: It is not possible to run a self-test on a channel, which has begun a data capture or generation cycle**.**

General “SYSTem” commands

SYSTem:PRESet

Description:
Reinitialize the module.

No Interrogative form

****

SYSTem:VERSion? ==> <num_val>

Description
Returns the SCPI revision number to which the instrument conforms.

****

SYSTem:ERRor? ==> <data_string>

Description:
Return an error message in the Error Output Queue, a FIFO queue. The oldest error message is sent first. , and error messages are removed from the queue once sent. When the queue is empty, the message "0, No error" is sent. If an error occurs after the queue is already full, the message "350, Queue overflow" is stored instead of the error. Any further errors are not stored.

****

SYSTem:CLOCk <INTernal | EXTernal>

Description:
Select the reference clock for the eight timers of the application cards.
Parameter:
INTernal: the reference clock is 10 MHz.
EXTernal: the reference clock is taken from the J5 connector on the front panel: 0 < External frequency < 16 MHz.

Interrogative form:
SYSTem:CLOCk? ==> <INTernal | EXTernal>

****

SYSTem:VOLTage:REFerence <INTernal | EXTernal>

Description:
Select the reference voltage for the application cards.
This voltage is taken from the J3 connector on the front panel.

Parameter:
INTernal: the reference voltage is 10V.
EXTernal: the reference voltage is taken on the J3 connector on the front panel.

Interrogative form:
SYSTem:VOLTage:REFerence? ==> <INTernal | EXTernal>

STATus Command

General Description

• The IEEE-488.2 standard requires two registers: Status Byte and Standard Event Status Register.

• The Status Byte provides global system status information that the user may select through the Enable Registers.

• The Standard Event Status Register reports standard events such as Error Detection or Operation Complete.

• SCPI has added a few supplementary registers, which represent the status and events specific to the instrument. Three registers are available at the same level:

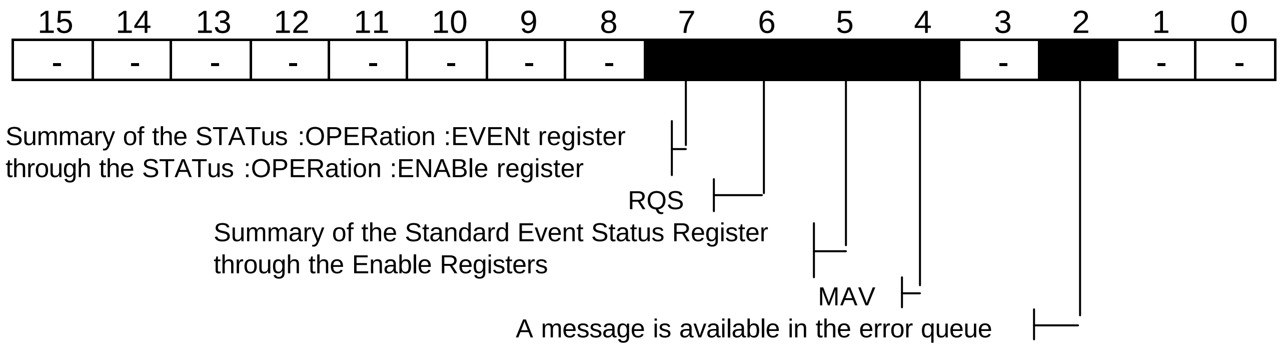
A **CONDition** register reports the state of the instrument, and each state change for the instrument is indicated in the Condition register.

An **EVENt** register captures changes in the associated Condition register. When a change has been captured, the Event register signals the change, even if the Condition register has returned to its initial state.

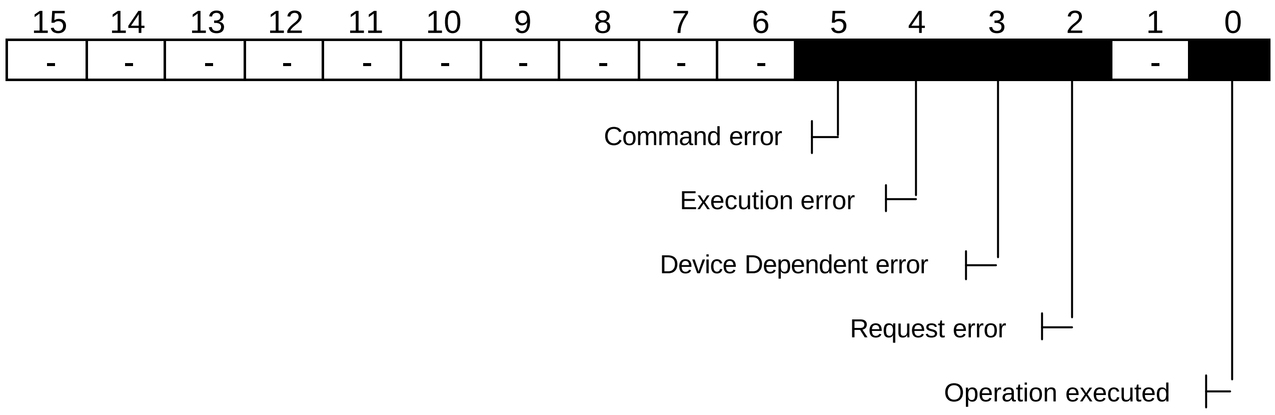
A few bits of the Event register do not reflect changes in the corresponding Condition register. These bits represent the state of the selected bits in other Event registers. The selection of these bits is specified in the **ENABle** register.

Description of the Status Register Contents

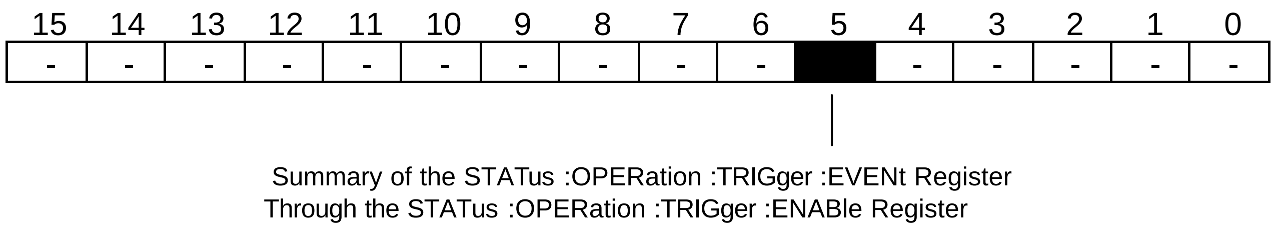
STATUS BYTE


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Summary of the STATus :OPERation :EVENt register
through the STATus :OPERation :ENABle register
RQS
Summary of the Standard Event Status Register
through the Enable Registers
MAV
A message is available in the error queue

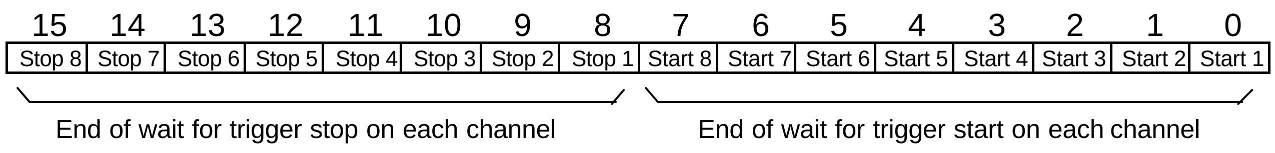
STANDARD EVENT STATUS REGISTER



STATus :OPERation :EVENt Register


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Summary of the STATus :OPERation :TRIGger :EVENt Register
Through the STATus :OPERation :ENABle Register

STATus :OPERation :TRIGger :EVENt Register


15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Stop 8 Stop 7 Stop 6 Stop 5 Stop 4 Stop 3 Stop 2 Stop 1 Start 8 Start 7 Start 6 Start 5 Start 4 Start 3 Start 2 Start 1
End of wait for trigger stop on each channel
End of wait for trigger start on each channel

STATus :OPERation :CONDition Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	■	-	-	-	-	-	-

Summary of the STATus :OPERation :TRIGger :EVENt Register
Through the STATus :OPERation :TRIGger :ENABle Register

STATus :OPERation :TRIGger :CONDition Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Stop 8	Stop 7	Stop 6	Stop 5	Stop 4	Stop 3	Stop 2	Stop 1	Start 8	Start 7	Start 6	Start 5	Start 4	Start 3	Start 2	Start 1

Wait for trigger stop on each channel

Wait for trigger start on each channel

SCPI STATus Commands

STATus:PRESet

Description:
Initializes all STATus ENABle registers, **but does not erase the EVENt registers**.

****

STATus:OPERation[:EVENt]? ==> <numeric_value>
STATus:OPERation:TRIGger[:EVENt]? ==> <numeric_value>

Description:
Return the contents of the specified Event register. Erases the contents of the specified Event register after reading.

WARNING:
This warning only concerns channels having a TTLTrig line or an external trigger as TRIGger STARt.
The bits reflecting the state of the channels (START or STOP) are copied in the STATus OPERation TRIGger EVENt register only if the SCPI "OUTPut:SIGNal:SOURce RUN_stop,<channel_list>" command has been sent (see the description of this command).
Example:
"TRIG:STAR:SOUR ETRG,(@1:8)"
"OUTP:SIGN:SOUR TIM,(@1:7)"
"OUTP:SIGN:SOUR RUN,(@8)" ==> only the states of channel 8 will be copied in bits 7 and 15 of the status register.

STATus:OPERation:CONDition? ==> <numeric_value>
STATus:OPERation:TRIGger:CONDition? ==> <numeric_value>

Description:
Return the contents of the specified CONDition register.

WARNING:
This warning only concerns channels having a TTLTrig line or an external trigger as TRIGger STARt.
The bits reflecting the state of the channels (START or STOP) are reported in the STATus OPERation TRIGger EVENt register only if the SCPI “OUTPut:SIGNal:SOURce RUN_stop,<channel_list>” command has been sent (see the description of this command).

Example:
“TRIG:STAR:SOUR TTLT0,(@1:8)”
“OUTP:SIGN:SOUR TIM,(@1:7)”
“OUTP:SIGN:SOUR RUN,(@8)” ==> only the state of channel 8 are reported in bits 7and 15 of the status register.

STATus:OPERation:ENABle <numeric_value>
STATus:OPERation:TRIGger:ENABle <numeric_value>

Description:
Set the specified ENABle register.

Initialization:
STATus:OPERation:ENABle is set to 0 so that no bits of the STATus:OPERation:EVENt register are reported to bit 7 of the Status Byte.

STATus:OPERation:TRIGger:ENABle is set to #HFFFF so that all the active bits of the STATus:OPERation:TRIGger:EVENt register are not reported in bit 5 of STATus:OPERation:EVENt.

Examples:
"STAT:OPER:ENAB 32" enables bit 5 of *STATus:OPERation:EVENT* to be reported in bit 7 of the Status Byte.
"FORM HEX;:STAT:OPER:TRIG:ENAB F000" enables the four high-order bits of *STATus:OPERation:TRIGger:EVENt* to be reported in bit 5 of *STATus:OPERation:EVENT* , in this case, the user wants to be told when channels 5, 6, 7 and 8 have finished their exchanges.

Interrogative form:
STATus:OPERation:ENABle? ==><numeric_value>
STATus:OPERation:TRIGger:ENABle? ==> <numeric_value>

****

Note *The ENABle registers only concern the EVENt registers. The status of the CONDition registers is determined by taking into account the ENABle registers.*

IEEE 488.2 Status Commands

***IDN? ==> RACAL,<Model_number>,<Serial_Number>,<Hardware_revision>**

Description:
Returns model identification.

****

***OPT? ==><data_string>**

Response:
==> "MEMOry" , if the memory option is installed.
==> "NONE" , if the memory option is not installed.
.

****

***TRG**

Description:
IEEE 488.2 command that generates a trigger equivalent to the GET command (Word Serial Trigger).

***CLS**

Description:
Erase all the Event registers and the Error Queue.

****

***ESE <numeric_value>**

Description:
Set the Standard Event Status Register to report in bit 5 of the Status Byte.

Initialization:
0, the Standard Event Status Register does not report in the Status Byte.

****

***ESE? ==> <numeric_value>**

Description:
Return the contents of the Standard Event Status Enable Register.

****

***ESR? ==> <numeric_value>**

Description:
Return the contents of the Standard Event Status Register (read then erase).

***STB? ==> <numeric_value>**

Description:
Return the contents of the Status Byte.

****

***RST**

Description:
Identical to the SCPI "SYSTem:PRESet" command.

****

***TST? ==> <data_string> | <numeric_value>**

Description:
Executes the self-test on all channels of the system and returns the result.

Response:

“PASSED” if the system passed the test.
"FAILED: <numeric_value> if the system failed the test..
<num_value>:
 *8: configuration error.
 *4: defective mass storage memory(if the option is installed)
 *2: error on application card 1
 *1: error on application card 2

Example:
 “*TST?” ==> 3 : error on both application cards.
 9 : configuration error + error on application card 2.

Note: It is not possible to run a self-test if one of the eight channels has begun a data capture or generation cycle**.**

Other valid commands:
***OPC, *OPC?, *SRE, *SRE? and *WAI**

Options

Option 03 (2.5 MS/s)

This option allows to increase the sampling frequency on the output channels until 2.5 MS/s instead of 500 KS/s.

Modified SCPI Command : **TRIGger[:SEQuence]:TIMer**

The period is always expressed in microseconds but the resolution is in hundred of nanoseconds.
The maximal period is 100000 µsec (100 milliseconds).
The minimal period is 0.4 µsec (400 nanoseconds).

Example :
"TRIG:TIM 0.5,(@5)" , the sampling period on channel 5 is 0.5µsec (500 nanoseconds).that corresponds to a frequency of 2MS/s.
"TRIG:TIM 10.1,(@5)" , the corresponding sampling frequency on channel 5 is 99 KS/s.

Option 04 (Filter)

This option allows to filter input or output channels. It is a low pass filter (8th order).

New SCPI commands.

**FILTer[:LPASs][:STATe]	<ON|OFF>,<channel_list>**

Description:

Turns the low pass filter ON or OFF on the selected channel.

When the filter is enabled, a new cut-off frequency is programmed. It corresponds to the sampling frequency divided by two of the selected channel. It is unnecessary to program the filter frequency by the « FILTer:FREQuency » command before enable the filter because this one will be recalculated.

Initialization : OFF

Example:

« TRIG:TIM 10,(@1) »;	sets the sampling frequency to 100kHz.
« FILT:FREQ 1000,(@1) »	sets the filter frequency to 1KHz.
« FILT ON,(@1) »,	enables the filter on channel 1 and programs again the filter frequency to 50KHz..

Interrogative form:

FILTer[:LPASs][:STATe]? <channel_list> ==>	<ON|OFF>{,<ON|OFF>,...}

****

FILTer[:LPASs]:SHAPe <LINear|ELLiptic>,<channel_list>

Description:

Determines the filter shape : LINear or ELLiptic.

Note: The filter frequency is not automatically modified by the filter shape change.

Initialization : ELLiptic

Interrogative form:

FILTer[:LPASs]:SHAPe? <channel_list> ==>	<LINear|ELLiptic>{,...}

FILTer[:LPASs]:FREQuency <numeric_value>,<channel_list>

Description:

Determine the cutoff frequency of the low pass filter.

Parameter:

<numeric_value> gives the frequency expressed in HERTZ in the <NRf> format of the IEEE 488.2 standard. If the value exceeds the upper or lower limits, an error is returned.

the extreme values depend on the filter shape :

ELLiptic shape:	min_val =	5 Hz
	max_val =	125000 Hz
LINear shape:	min_val =	5 Hz
	max_val =	62500 Hz

Initialization : 0

Examples:

LINear shape :
« FILT:SHAP LINear,(@1) », sets the linear filter.
« FILT:FREQ 1017.5,(@1) » sets the cutoff frequency to 1017.5 Hz.
« FILT:FREQ? (@1) », the reel programmed frequency is **1016.260** Hz.

ELLiptic shape :
« FILT:SHAP ELLiptic,(@1) », sets the elliptic filter.
« FILT:FREQ 1017.5,(@1) » sets the cutoff frequency to 1017.5 Hz.
« FILT:FREQ? (@1) », the reel programmed frequency is **1000.000** Hz.

Interrogative form:

FILTer[:LPASs]:FREQuency? <channel_list> ==> <numeric_value>
Return the actually programmed cutoff frequency.

Warning: If the filter is enabled (ON), an automatic adjustment is operated at every new sampling period setting (« TRIGger:TIMer <period>,<channel_list>»). To program the filter frequency, the user will make sure that the « FILT:FREQ » command after the sampling period setting is sent.

Option 05 (250 KS/s)

This option allows to increase the sampling frequency on the input channels until 250 KS/s instead of 100 KS/s.

Modified SCPI Command : **TRIGger[:SEQuence]:TIMer**

The period is always expressed in microseconds but the resolution is in hundred of nanoseconds.
The maximal period is 100000µsec (100 milliseconds).
The minimal period is.4 µsec.

Example :
"TRIG:TIM 4.2,(@1)" , , the corresponding sampling frequency on channel 1 is 238 KS/s.

Option 06 (Isolated amplifier with programmable gain)

New SCPI commands :

CORRection:GAIN[:LEVel]   <numeric_value>,<channel_list>

Description:

Allow to control gain on selected channel by <channel_list>.

Parameter:

<numeric_value> gives the gain value to apply on the selected output channel. It is expressed in the <NRf> format of the IEEE 488.2 standard. The value is from 0 and 2.50.

The resolution is $6.1*10^{-4}$.

Initialization :   1

Example:

"VOLT 10,(@1)"	the analog output is set to 10V.
"CORR:GAIN 1.0,(@1)"	sets gain to 1 on channel 1 => 10V
"CORR:GAIN 2.42,(@1)"	sets gain to 2.42 on channel 1 => 24.2V

Interrogative form:

CORRection:GAIN[:LEVel]? <channel_list>   ==>   <numeric_value>

****

CORRection:GAIN:DATA   *<numeric_val>,<channel_list>

Description:

Allow to control gain on selected channel by <channel_list>.

Unlike the previous LEVel command, this command allows direct control of the digital/analog converter (gain controller 12 bits).

Parameter:

<numeric_value> is an integer value from 0 to #HFFF (0 to 4095).

Conversion table:

<numeric_value>	->	gain
#H0		0.0
#H800		1.25
#HFFF		2.50

Example:

"CORR:GAIN:DATA #H666,(@1)" sets gain to 1 on channel 1.

Interrogative form:

CORRection:GAIN:DATA? <channel_list> ==> *<numeric_value>

Appendix

Error Messages

No Errors

Error Description [Description/Explanation/Examples]
Number

0 No Errors
[The queue is empty. Each error/event in the queue has been read or the queue was erased by power on.

Command Errors

Error Description [Description/Explanation/Examples]
Number

-101 Invalid character
[A element of the syntax contains a character invalid for this type]

-102 Syntax error
[Command not recognized, or incorrect data type found. For example, a character string was received when the instrument expected something else.]

-108 Illegal Parameter
[The number of parameters is greater than expected. For example, the *SRE command only accepts one parameter, so the command *SRE 180,34 is illegal.]

-109 Missing Parameter
[The number of parameters is less than required. For example, the *SRE command requires a parameter, so the command *SRE is illegal.]

-112 Command mnemonic too long
[The command mnemonic contains at most twelve characters (see IEEE 488.2, 7.6.1.4.1).]

-113 Unknown Command
[The command syntax is correct, but the command is not defined for this instrument. For example, *XYZ is defined for no instruments.]

-120 Numeric data error
[Like Error -121, this error is generated when data appears that includes non-decimal numeric data types.
This error message may appear if the instrument cannot detect a more specific error]
or
Invalid Data character
[An invalid character has been received in a <data_string>.]

-121 Invalid character in number
[An invalid data type character has been received. For example, an alphabetic character in a number.]

-144 Data string too long
[The <data_string> element contains more than twelve characters (see IEEE 488.2, 7.7.1.4).]

-161 Invalid block data
[A block data element was expected, but was invalid for some reason (see IEEE488.2,7.7.6.2); for example, an END message was received before the length was satisfied ; the block data is too short or too long]

Execution Errors

Error Description [Description/Explanation/Examples]
Number

-200 Execution Error
[This error occurs when the TRACE DEPTH is UNDEFINED or
if the INTERRUPT QUEUE is FULL]

-210 Trigger error

-211 Trigger ignored
[A GET, *TRG or trigger signal has been received and recognized by the instrument but, for timing considerations, has been ignored. For example, the instrument is not ready to respond.]

-213 Initialization ignored
[Indicates an Initialization command has been ignored because another is already in progress.]

-221 Configuration conflict
[A command has been interpreted but cannot be executed because of the current state of the instrument.]

-222 Data out of bounds
[A command has been interpreted but cannot be executed because its value is out of bounds.]

-223 Too much data
[A block command, an expression or a character string has been received that contains more data than the instrument can accept because of its memory characteristics or any other specific characteristic of the instrument.]

-224 Illegal Parameter Value
[Used when a certain value, when possible, is expected. For example, the “TRIG:BUS HOLD” command generates an Error because the interpreter expects an ON or OFF parameter instead of HOLD.]

-241 hardware missing
[A command or request may not be executed because the corresponding hardware does not exist. For example, an option is not installed.]

-291 Out of memory

Instrument-specific Errors

Error Description [Description/Explanation/Examples]
Number

-300 Instrument-specific Error
[An error specific to the instrument, as defined in IEEE 488.2 11.5.1.1.6 has occurred. For example, the self-test has been aborted.]

-311 Memory-related error
[An error has occurred in one of the memory areas of the module. The problem may arise when programming FLASH memory or when transferring samples in DMA mode.]

-350 Queue Overflow
[A specific code has been entered in the Queue instead of the code, which caused the error. This code indicates that the Queue is full and that the error that occurred has not been recorded.]

VXI Bus Word Serial commands

The module supports the following commands:

- Abort Normal Operation
- Assign Interrupter Line
- Asynchronous Mode Control
- Begin Normal Operation
- Byte Available
- Byte Request
- Clear
- Control Event
- End Normal Operation
- Read Interrupter Line
- Read Interrupters
- Read Protocol
- Read Protocol Error
- Read STB
- Trigger

Abort Normal Operation

This command is used to halt normal operations. When the instrument receives this command, it returns to its default configuration, aborting any ongoing operations. The aborted state is defined as follows: waiting interrupts are not acknowledged, new bus or interrupt requests may not be acknowledged. The instrument is in an inactive state and is ready to receive commands.

Command:
$C8FF_{16}$

The response is placed in the Data Low Register.
Response:
$FFFE_{16}$

Assigning interrupt lines

This command is used to assign an IRQ line of the VME bus to a slave interrupt generator.

Command:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	0	1	0	1	0	X	Int_ID			X	Line		

• X: Insignificant
• Int_ID: Unique identifier of the assigned interrupt generator. For this model, the 'INT_ID' field has the value '001'.
• Line: Number of the IRQ line on the VME bus. A zero value indicates the interrupt generator is disconnected.

Action:
if Int_ID = 001, save the value of 'Line'

The response is placed in the Data Low Register.
Response:
if Int_ID = 001 --> $FFFE_{16}$
else --> $7FFE_{16}$

Asynchronous control mode

A «master» to create a path for events and responses uses this command.

Command:

15	14	13	12	11	10	9	8	7 6 5 4	3	2	1	0
1	0	1	0	0	0	1	0	X	Val Rep*	Val Event*	Rep Mode	Event Mode

• X: Insignificant
• Val Resp*: a zero (0) enables reply generation. A one (1) disables reply generation.
• Val Event*: a zero (0) enables event generation. A one (1) disables event generation.
• Resp.Mode: A one (1) indicates responses may be sent as signals. A zero (0) indicates responses may be sent as interrupts.
• Event Mode: A one (1) indicates events may be sent as signals. A zero (0) indicates events may be sent as interrupts.

The response is place in the Data Low Register.
Response:

15 14 13 12	11	10	9	8	7	6	5	4	3	2	1	0
Status	1	1	1	1	1	1	1	1	Val Rep*	Val Event*	Rep Mode	Event Mode

• Status: This field indicates the status of command execution.
- F_{16}: The command has executed correctly.
- 7_{16}: The command did not execute. A required option was not supported.
• Val Resp*: a zero (0) enables reply generation. A one (1) disables reply generation.
• Val Event*: a zero (0) enables event generation. A one (1) disables event generation.
• Resp.Mode: A one (1) indicates responses may be sent as signals. A zero (0) indicates responses may be sent as interrupts.
• Event Mode: A one (1) indicates events may be sent as signals. A zero (0) indicates events may be sent as interrupts.

Begin Normal Operation

This command tells the instrument it may execute a normal operational cycle.

Command:
$FCFF_{16}$ or $FDFE_{16}$

The response is placed in the Data Low Register.
Response:
$FFFE_{16}$

Byte Available

A «master» to send a byte of data to the «slave» uses this command. The END field indicates the last byte of the message.

Command:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	1	1	1	0	END	DATA							

Response:
no response

Byte request

The «master» to receive a data byte from a «slave» instrument uses this command.

Command:
$AEFF_{16}$

The response is placed in the Data Low Register. The END field indicates the last byte of the message.
Response:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	1	1	1	0	END	DATA							

Clear

This command is used by a "Master" to force a "slave" to re-initialize the VXI interface and interrupt all operations. Any operation underway when this command is received is terminated normally.

Command:
$FFFF_{16}$

Response:
no response

Control event

This command is used by a "Master" to selectively enable events generated by the "slave". A one (1) in the validation field enables specific event generation. A zero (0) in the validation field disables specific event generation.

Command:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	0	1	1	1	1	Enable	Event						

• Event : These bits (6 <- 0) identify the bits (14 <- 8) of the enabled/disabled event. See Section E.4, "Protocol Events", for more information.

The response is placed in the Data Low Register.
Responses:
- $FFFE_{16}$: the command terminated normally.
- $7FFE_{16}$: command default. The instrument does not generate the referenced event.

End Normal Operation

This command is used by the "Master" to methodically halt all operations. Upon receipt of this command, the "Master" sends an End Normal Operation command *to all message-based "slaves".* The "Master" is responsible for halting the "slaves". The "End" state is defined as follows: waiting interrupts are not acknowledged, new bus or interrupt requests may not be acknowledged. The instrument is in an inactive state, and is ready to receive other commands.

Command:
$C9FF_{16}$

The response is placed in the Data Low Register.
Response:
$FFFE_{16}$

Read Interrupt Line

This command is used to determine to which of the VME bus' IRQ lines a slave's particular interrupt generator is connected.

Command:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	1	0	1	X					Int_ID		

• X: Insignificant
• Int_ID: Unique identifier of the specific interrupt generator. For this module the 'Int_ID' field must contain the value '001'.

The VME bus' IRQ line number is placed in the Data Low Register.

Response:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Status				1	1	1	1	1	1	1	1	1	Line		

• Status : This field indicates the status of command execution.
 - F_{16}: The command has executed correctly.
 - 7_{16}: Command default. The interrupt generator referenced in the Int_ID field is unknown to the instrument..
• Line: The IRQ line number on the VME bus. A value of zero (0) indicates the interrupt generator is disconnected.

Read Interrupts

This command is used to determine the number of a "slave’s interrupt generator.

Command:
CAFF$_{16}$

The number of the interrupt generator is placed in the Data Low Register.

Response:
FFF9$_{16}$ (an interrupt generator).

Read Protocol

A «master» to know what protocol is used in the word serial protocol used by the «slave» uses this command.

Command:
DFFF$_{16}$

The supported protocol word is placed in the Data Low Register.
Response:
FE23$_{16}$

This instrument :
- is capable of generating the events.
- supports the *Read Interrupts*, *Read Interrupt Line*, and *Assign Interrupt Line* commands.
- supports the Word Serial Trigger commands.
- supports the VXI IEEE-488.2 Instrument protocol.
- supports the VXI Instrument protocol..

Read Protocol Error

A «master» to ask its «slave» to report its current error status, in the form of a 16-bit response uses this command. After having replied to this command, the "slave" resets its Current Error status to "No Error".

Command:
$CDFF_{16}$

The error code is placed in the Data Low Register.
There are many possible responses:

No Error: $FFFF_{16}$

Multiple Queries: $FFFD_{16}$
The instrument was instructed to write over a previous unread response.

Unsupported command: $FFFC_{16}$
The instrument received an unsupported command.

DIR violation: $FFFB_{16}$
The instrument received a command, which violates the DIR protocol.

DOR violation: $FFFA_{16}$
The instrument received a command, which violates the DOR protocol.

Read Ready violation: $FFF9_{16}$
The instrument received a command, which violates the Read Ready protocol.

Write Ready violation: $FFF8_{16}$
The instrument received a command, which violates the Write Ready protocol.

Read STB

This command allows a "master" to read the status byte of a "slave".

Command:
CFFF$_{16}$

The status byte is placed in the Data Low Register.

Response:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	1	Status Byte							

Trigger

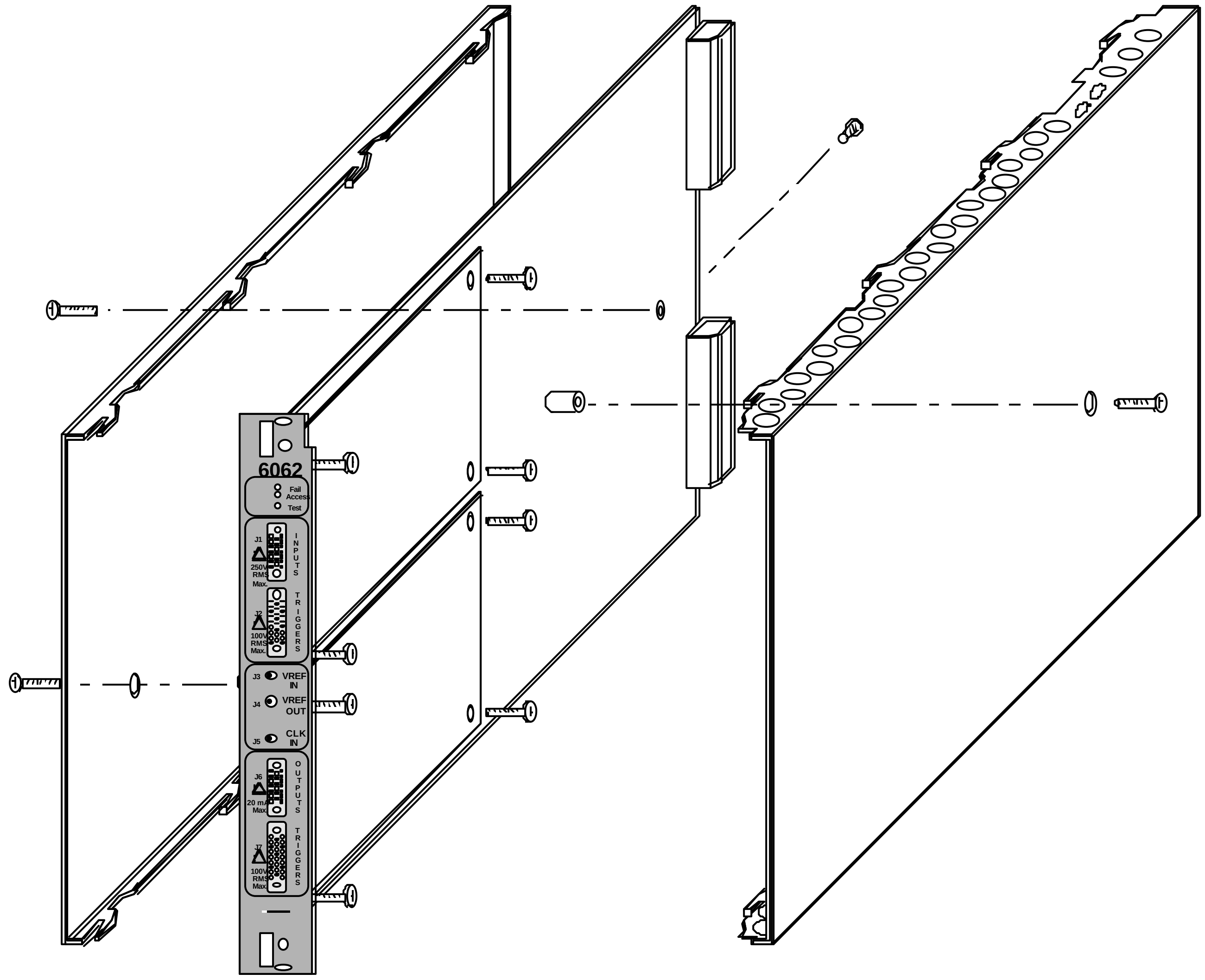
A «master» to ask a «slave» to trigger a previously armed operation uses this command.

Command:
EDFF$_{16}$

Response:
No response

6. Calibrating the module

Opening the module



In order to disassemble the electronic boards from the upper and lower covers, you must unscrew the two screws on the lower cover, the screw on the top cover, and the four screws behind both covers. Then you must slide the two covers towards the rear while holding the module's front panel.

Warning You must not touch the boards unless you are wearing an anti-static bracelet. If not you may destroy some sensitive components.

All adjustments and test points are accessible on the topside of the electronic boards.

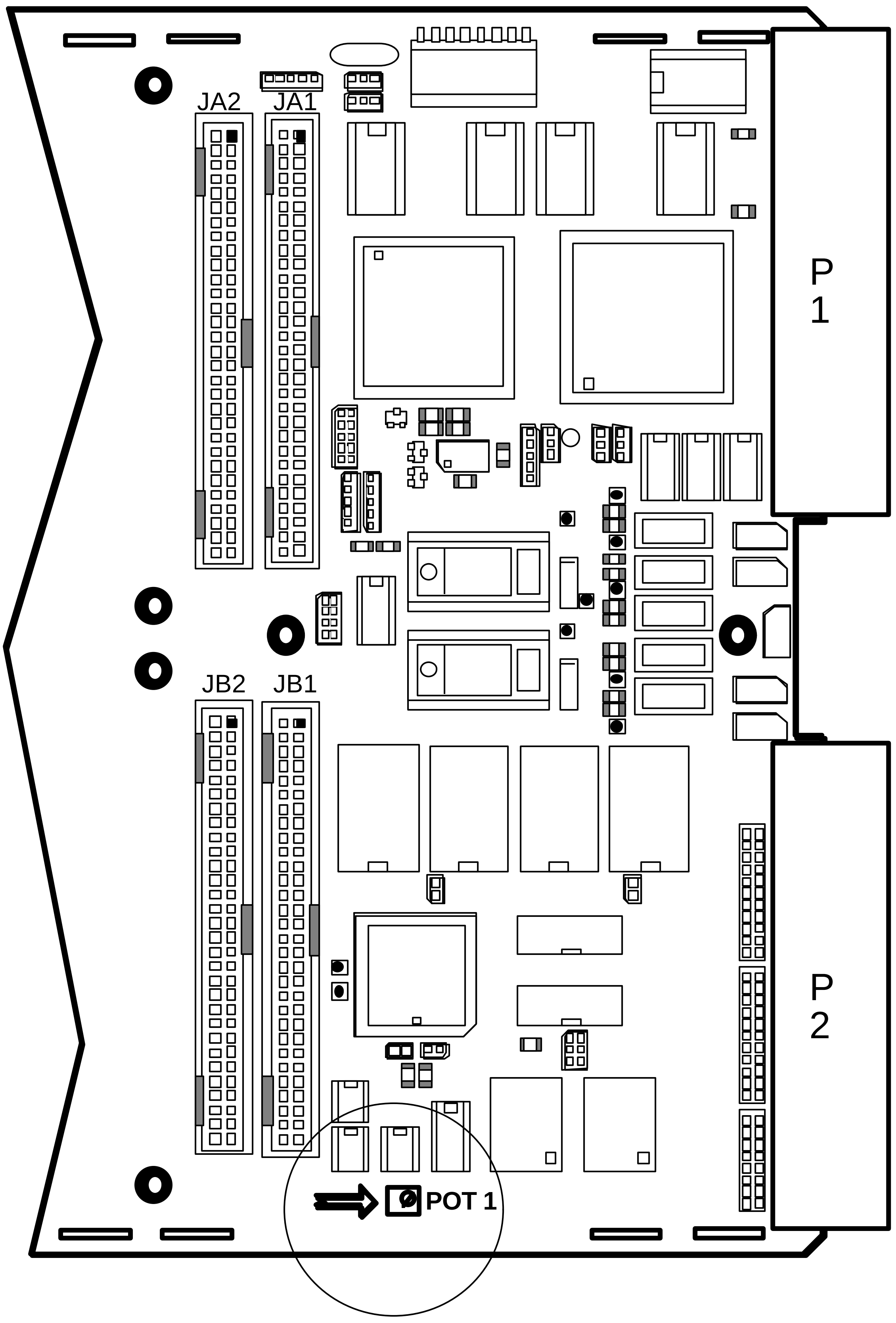
Motherboard

Material required

- A continuous voltmeter with at least 5 $^{1/2}$ Digits.
- A coaxial / banana cable (coaxial plug supplies)
- A small screwdriver.

Adjusting the 10.000V reference voltage

- Insert the module in a free slot in the VXI chassis.
- Power on the VXI chassis.
- Let the module stabilize for 10 minutes at room temperature (23°C ± 2°C).
- Connect the VREF OUT output (J4) to the voltmeter (position DC, range 20V), using the coaxial cable.
- Adjust the POT1 potentiometer on the motherboard so the voltmeter reads 10.000V ±1mV.



Analog input boards

Material required

- A continuous voltmeter with at least 5 $^{1/2}$ Digits.

- A continuous ±10V calibrator, with a precision of 153µV (1/2 LSB).
- A BF 1Hz - 1MHz generator, 20V peak to peak
- A BF millivoltmeter, 20Hz - 20MHz, with full-scale precision of 10^{-4}.
- A 4 twisted-pair shielded cable (Positronic 20-pin plug provided)
- A high impedance BNC probe
- 2 BNC measurement Ts
- A BNC 50Ω resistance
- A small screwdriver.

Adjusting the input filter

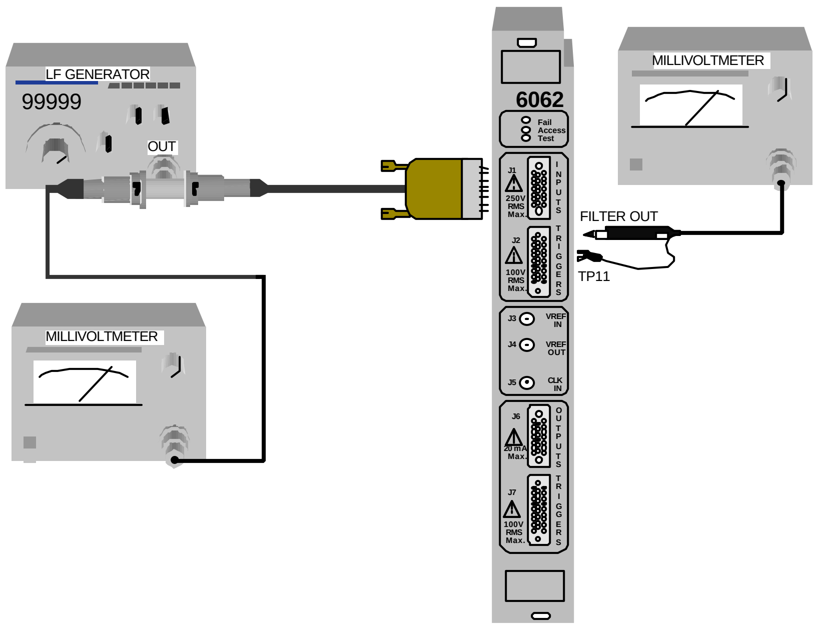
- Remove the two upper shielding covers using a small screwdriver, using it as a lever on the input/output plug holding screws.

- Insert the module in a free slot of the VXI chassis.

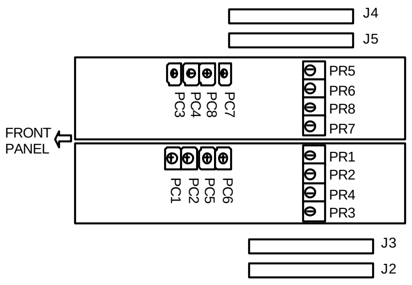
- Power on the VXI chassis.

- Let the module stabilize for 10 minutes at room temperature (23°C ± 2°C).

- Connect the channel 1 input as shown in the figure below.



• Connect the high impedance BNC probe to the output filter (contact point on pin X of J2, ground on TP13). The BF millivoltmeter may be used for both measurements.



• Supply a 25kHz, ±10V square signal to the channel 1 input.

• Program the sampling frequency to 50K samples/ second.

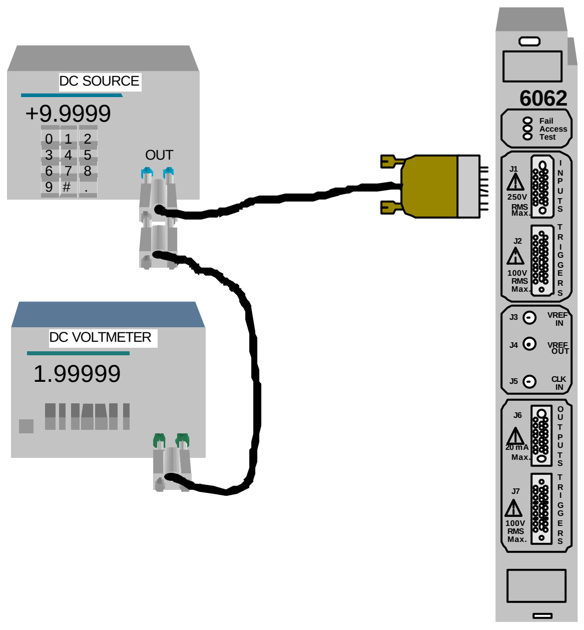
• Adjust PC1 and PC5 to obtain a slew rate of 5V / µs (minimum overshoot).

• Follow the same procedure for the other channels. The table below provides the corresponding adjustments for each input channel.

Channel	**Output filter**	**1st adjustment**	**2nd adjustment**
1	J2, pin X	PC1	PC5
2	J3, pin X	PC2	PC6
3	J4, pin X	PC3	PC7
4	J5, pin X	PC4	PC8

Adjusting the offset and gain voltage

• Connect the calibrator input as shown in the following diagram.



1) Initialize the four analog inputs by issuing the following commands:

- TRIG:TIM 10, (@1:4)	(sample every 10µs)
- MEM:LENGth 20,(@1:4)	(measure over 20 samples)
- INIT (@1:4)	(start the four channels)

2) Apply a short circuit to input 1 (IN1+ = IN1- = GROUND).

3) Acquire the 20 samples by issuing the following command:

- MEM:DATA? 1,HEX	(read the 20 samples on channel 1)

4) Adjust the PR1 potentiometer to obtain values between **FFFE**$_{HEX}$ and **0002**$_{HEX}$ (repeat steps 3 and 4 several times).

5) Apply 9.9997V ± 0.15mV to input 1.

6) Acquire 20 samples by issuing the following command:
- MEM:DATA? 1,HEX (read 20 samples on channel 1)

7) Adjust the PR2 potentiometer to obtain values between **7FFC**$_{HEX}$ and **7FFF**$_{HEX}$ (repeat steps 6 and 7several times).

8) Apply -10.000V ±0.15mV to input 1.

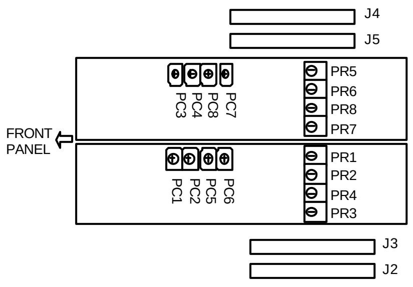
9) Acquire 20 samples by issuing the following command:
- MEM:DATA? 1,HEX (read 20 samples on channel 1)

10) Check to make sure if the values read are between **8004_{HEX}** and **8000_{HEX}**. if the values are outside this range, start the adjustments again from step 2.

Repeat steps 2 through 10 for the other channels. In this case, the command to read samples is:
- MEM:DATA? X,HEX where **X** is the number of the channel to sample (1 to 4).

Channel	Offset	Gain
1	PR1	PR2
2	PR3	PR4
3	PR5	PR6
4	PR7	PR8

Refer to the figure below for the positions of the adjustment components:



Analog output boards

Material required

- A continuous voltmeter with at least 5 $^{1/2}$ Digits.
- A BF 1Hz - 1MHz generator, 20V peak to peak
- A BF millivoltmeter, 20Hz - 20MHz, with full-scale precision of 10^{-4}.
- A 4 twisted-pair shielded cable (Positronic 20-pin plug provided)
- A high impedance BNC probe
- 2 BNC measurement Ts
- A BNC 50Ω resistance
- A small screwdriver.

Adjusting the output filter

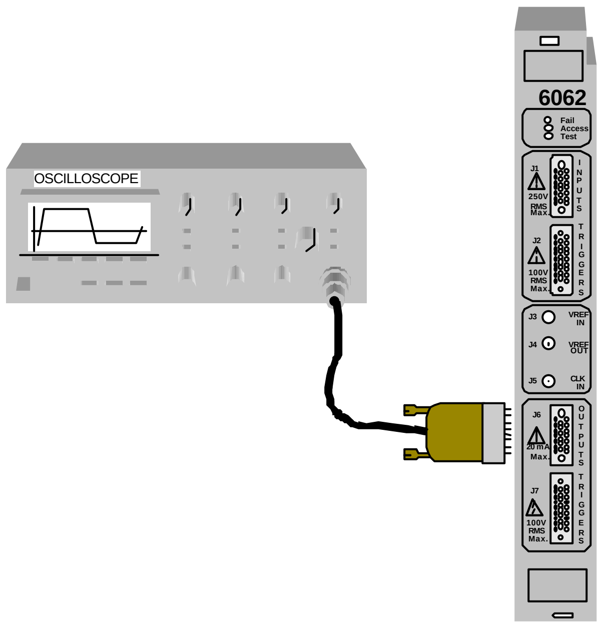
- Remove the two upper shielding covers using a small screwdriver, using it as a lever on the input/output plug holding screws.

- Insert the module in a free slot of the VXI chassis.

- Power on the VXI chassis.

- Let the module stabilize for 10 minutes at room temperature (23°C ± 2°C).

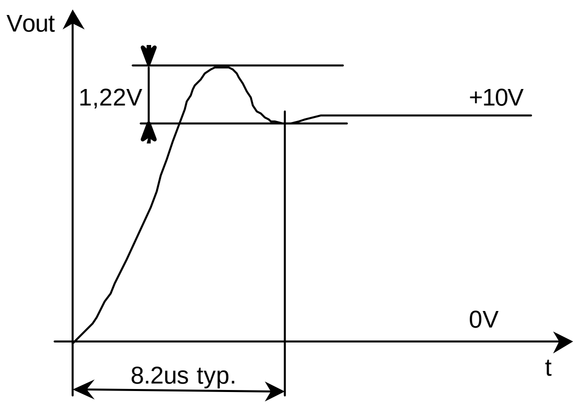
- Connect the channel 5 output as shown in the figure below.



- The following commands generate a square signal with ±10V amplitude and a frequency of 25kHz.

- **OUTP ON,(@5:8)**	(activate output relays)
- TRIG:TIM 20, (@5:8)	(sample period is 40µs)
- MEM:LENGth 2,(@5:8)	(measure 2 samples)
- MEM:DATA 5,32768,32767	(2 samples -10V and +10V on channel 5)
- MEM:DATA 6,32768,32767	(2 samples -10V and +10V on channel 6)
- MEM:DATA 7,32768,32767	(2 samples -10V and +10V on channel 7)
- MEM:DATA 8,32768,32767	(2 samples -10V and +10V on channel 8)
- INIT (@5:8)	(start the four channels)

- Display on the oscilloscope the voltage when going from -10V / +10V.
- Adjust PC9 to obtain a maximum overshoot voltage.
- Adjust PC1 to obtain a minimum overshoot voltage.
- Adjust PC5 to obtain a minimum overshoot voltage.
- Adjust PC9 to obtain an overshoot voltage equal to 1.22V +/- 10%

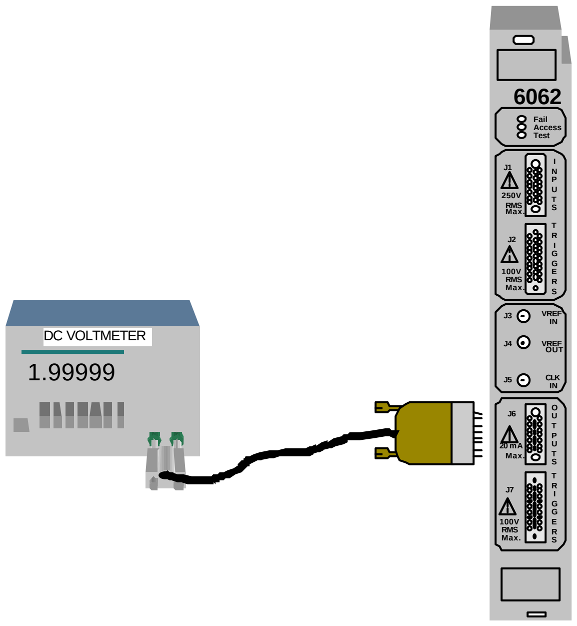


• Follow the same procedure for the other channels. The following table provides the corresponding adjustments for each input channel.

Channel	1st and 4th adjustments	2nd adjustment	3rd adjustment
5	PC9	PC1	PC5
6	PC10	PC2	PC6
7	PC11	PC3	PC7
8	PC12	PC4	PC8

Adjusting the offset and gain voltage

• Connect the calibrator input as shown in the following diagram.



1) Initialize the four analog outputs by issuing the following command:
- **OUTP ON,(@5:8)** (activate the output relays)

2) Set output voltage to 0V by issuing the following command:
- VOLT:DATA #H0000, (@5)

4) Adjust the PR1 potentiometer to measure 0V ±0.3mV.

5) Set output voltage to 9.9997V by issuing the following command:
- VOLT:DATA #H7FFF, (@5)

6) Adjust the PR2 potentiometer to measure 9.9997V ±0.3mV.

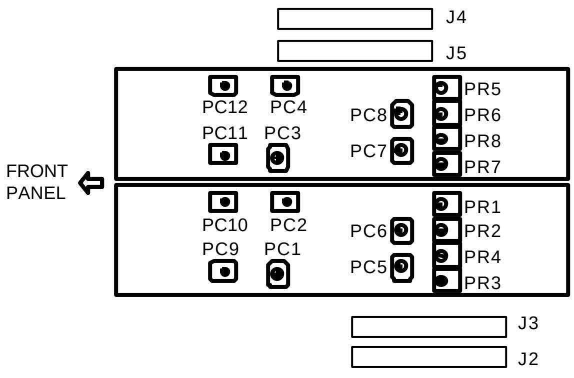
7) Set output voltage to -10.000V by issuing the following command:
- VOLT:DATA #H8000, (@5)

8) Check that the output voltage is equal to -10.000V ±0.3mV. If this is not the case, start again at step 2.

Repeat steps 2 through 10 for each channel. In this case, the command changes to:
- VOLT:DATA #HXXXX, (@Y) where **XXXX** is the hexadecimal value and Y is the number of the channel to sample (5 to 8).

Channel	Offset	Gain
5	PR1	PR2
6	PR3	PR4
7	PR5	PR6
8	PR7	PR8

Refer to the figure below for the layout of the adjustment components.



Option 06 and option 06-S2 boards

Material required

(TE 1)Calibrator FLUKE 5100B or equivalent,
(TE 2)Voltmeter KEITHLEY 2001 or equivalent,
VXI chassis,
PC with MXI1 or 2 link,
RESMAN Software
VIC Software.

Option 06 Configuration

Not fitted components :

-Jumper : R26, R56, R58, R60, R62, R64, R66, R70, R72, R74, R76, R78, R80, R82, R85, R87, R89, R90

Component values :

- Resistor :	R29, R35, R41, R47	215kΩ 0.1% 15ppm/°C
	R30, R36, R42, R48	44.2kΩ 0.1% 15ppm/°C
	R27, R28, R33, R34, R39, R40, R45, R46	51.1kΩ 0.1% 15ppm/°C

Option 06 calibration

Offset adjustment

Commands to send to the module.

Module version and daughter board	Commands	Result
6062 B4 Opt 06	CORR:GAIN 2 ,(@1:4)	Gain = 2 for channels 1 to 4

Apply a 0V with the calibrator at the option 06 input.

Board	Channel	Input	Voltmeter +	Voltmeter -	Adjustable resistor	Voltage	Limits
2	1	IN1	PT1	PT2	PR1	0V	± 1mV
2	1	IN1	PT4	PT3	PR5	0V	± 1mV
2	2	IN2	PT5	PT6	PR2	0V	± 1mV
2	2	IN2	PT8	PT7	PR6	0V	± 1mV
2	3	IN3	PT9	PT10	PR3	0V	± 1mV
2	3	IN3	PT20	PT19	PR7	0V	± 1mV
2	4	IN4	PT21	PT22	PR4	0V	± 1mV
2	4	IN4	PT24	PT23	PR8	0V	± 1mV

Gain adjustment

Commands to send to the module.

Module version and daughter board	Commands	Result
6062 B4 Opt 06	CORR:GAIN 2 ,(@1:4)	Gain = 2 for channels 1 to 4

Apply a +10V with the calibrator at the option 06 input.

Board	Channel	Input	Voltmeter +	Voltmeter -	Adjustable resistor	Voltage	Limits
2	1	IN1	Y(OUT1+)	X(GND1)	PR12	+10V	± 2mV
2	1	IN1	W(OUT1-)	X(GND1)	PR11	-10V	± 2mV
2	2	IN2	S(OUT2+)	R(GND2)	PR14	+10V	± 2mV
2	2	IN2	P(OUT2-)	R(GND2)	PR13	-10V	± 2mV
2	3	IN3	K(OUT3+)	J(GND3)	PR18	+10V	± 2mV
2	3	IN3	H(OUT3-)	J(GND3)	PR17	-10V	± 2mV
2	4	IN4	C(OUT4+)	B(GND4)	PR16	+10V	± 2mV
2	4	IN4	A(OUT4-)	B(GND4)	PR15	-10V	± 2mV

Option 06-S2 Configuration

Not fitted components :

-Jumper : R25, R55, R57, R59, R61, R63, R65, R69, R71, R73, R75, R77, R79, R81, R86, R88
-Capacitor : C25, C26, C37, C38, C104, C105, C110, C111

Component values :

- Resistor :	R29, R35, R41, R47	53.6kΩ 0.1% 15ppm/°C
	R30, R36, R42, R48	26.1kΩ 0.1% 15ppm/°C
	R27, R28, R33, R34, R39, R40, R45, R46	51.1kΩ 0.1% 15ppm/°C

Option 06-S2 calibration

Offset adjustment

Commands to send to the module.

Module version and daughter board	Commands	Result
6062 B4 Opt 06	CORR:GAIN 2.5 ,(@1:4)	Gain = 1.6x2.5 for channels 1 to 4

Apply a 0V with the calibrator at the option 06 input.

Board	Channel	Input	Voltmeter +	Voltmeter -	Adjustable resistor	Voltage	Limits
2	1	IN1	PT1	PT2	PR1	0V	± 1mV
2	1	IN1	PT4	PT3	PR5	0V	± 1mV
2	2	IN2	PT5	PT6	PR2	0V	± 1mV
2	2	IN2	PT8	PT7	PR6	0V	± 1mV
2	3	IN3	PT9	PT10	PR3	0V	± 1mV
2	3	IN3	PT20	PT19	PR7	0V	± 1mV
2	4	IN4	PT21	PT22	PR4	0V	± 1mV
2	4	IN4	PT24	PT23	PR8	0V	± 1mV

Gain adjustment

Commands to send to the module.

Module version and daughter board	Commands	Result
6062 B4 Opt 06	CORR:GAIN 2.5 ,(@1:4)	Gain = 1.6x2.5 for channels 1 to 4

Apply a +10V with the calibrator at the option 06 input.

Board	Channel	Input	Voltmeter +	Voltmeter -	Adjustable resistor	Voltage	Limits
2	1	IN1	Y(OUT1+)	X(GND1)	PR12	+20V	± 2mV
2	1	IN1	W(OUT1-)	X(GND1)	PR11	-20V	± 2mV
2	2	IN2	S(OUT2+)	R(GND2)	PR14	+20V	± 2mV
2	2	IN2	P(OUT2-)	R(GND2)	PR13	-20V	± 2mV
2	3	IN3	K(OUT3+)	J(GND3)	PR18	+20V	± 2mV
2	3	IN3	H(OUT3-)	J(GND3)	PR17	-20V	± 2mV
2	4	IN4	C(OUT4+)	B(GND4)	PR16	+20V	± 2mV
2	4	IN4	A(OUT4-)	B(GND4)	PR15	-20V	± 2mV

Calibration test program for the module

The program listed below is an example written in Quick BASIC 4.5 which performs the functional testing and calibration for the 6062A module.
This program was designed to work with slot 0 driven by a GPIB link such as the RACAL INSTRUMENT 1260-00B.
The initialization, read and write routines should be completed according to the type of GPIB card installed in the host machine. Do not forget to link the library supplied with the GPIB card to the Quick BASIC 4.5 library.

```
DECLARE SUB choice (v%)
DEFINT A-Z

'=============================================================
'                    6062 TEST Program
'                    Language :     QuickBasic 4.5
'                    IBM PC + Interface IEEE-488 INES
'                    Version : 1.0   Created : 24/2/1995
'                    Modified  : 27/3/1995
'=============================================================


'=============================================================
'            VARIABLE DECLARATIONS
'=============================================================

' GPIB card variables

DEFINT A-Z
r$ = SPACE$(80)


'=============================================================
'            MAIN PROGRAM
'=============================================================

SCREEN 9
CLS


'=============================================================
'             Initialize the  GPIB interface
'=============================================================
'=============================================================
'             Main loop
'=============================================================

r$ = "not available"
beginning:
```

```
CLS
LOCATE 4, 28: PRINT "Self-test : "; r$


COLOR 3
LOCATE 1, 20: PRINT "CONTROL OF THE 6062 MODULE, Version 1.0"
COLOR 7

m$ = "SYST:PRES"
CALL WRITE

m$ = "*IDN?"
CALL WRITE
CALL READ
LOCATE 3, 28: PRINT "Identification : "; r$

COLOR 15
LOCATE 19, 11: PRINT "Do not forget to adjust Vrefout (POT1 on MB for 10V
   ±0.3mV)"
```

```
COLOR 7

LOCATE 7, 15: PRINT "Output filter adjustments (1)"
LOCATE 10, 15: PRINT "Analog output adjustments (2)"
LOCATE 13, 15: PRINT " Input filter adjustments(3)"
LOCATE 16, 15: PRINT " Analog input adjustments(4)"
LOCATE 22, 15: PRINT "Quit (Q)"

DO

CALL choice(v%)

LOOP WHILE v% = 0

SELECT CASE v%
   CASE 49
   GOTO label1
   CASE 50
   GOTO label2
   CASE 51
   GOTO label3
   CASE 52
   GOTO label4
   CASE ELSE

   CLS
   LOCATE 20, 15: PRINT "Do you really want to quit (Y/N) ?"
   DO
   a$ = UCASE$(INKEY$)
   LOOP WHILE a$ = ""
   IF a$ = "Y" THEN
      CLS
      END
   ELSE
      r$ = " not available"
      GOTO beginning
   END IF
END SELECT


'===========================================================
'              Adjust output filters
'===========================================================

label1:
CLS

FOR i% = 1 TO 4

LOCATE 2, 20: PRINT "ADJUST OUTPUT FILTERS"
```

```
m$ = "OUTP ON,(@5:8)"
CALL WRITE

m$ = "TRIG:TIM 20,(@5:8)"
CALL WRITE

m$ = "MEM:LENGTH 2,(@5:8)"
CALL WRITE

m$ = "MEM:DATA 5,32768,32767"
CALL WRITE

m$ = "MEM:DATA 6,32768,32767"
CALL WRITE

m$ = "MEM:DATA 7,32768,32767"
CALL WRITE

m$ = "MEM:DATA 8,32768,32767"
CALL WRITE

m$ = "INIT (@5:8)"
CALL WRITE

LOCATE 12, 5: PRINT "Connect the scope to output "; STR$(i% + 4);
LOCATE 13, 5: PRINT "The output will generate a square signal at +/-10V at
   25kHz"
LOCATE 14, 5: PRINT "Adjust PC"; STR$(i% + 8); " for maximum overshoot"
CALL following
LOCATE 15, 5: PRINT " Adjust PC"; STR$(i%); " for maximum overshoot "
CALL following
LOCATE 16, 5: PRINT " Adjust PC"; STR$(i% + 4); " for maximum overshoot "
CALL following
LOCATE 17, 5: PRINT " Adjust PC"; STR$(i% + 8); " for an overshoot of 1.22V"
CALL following
LOCATE 18, 5: PRINT "Verify the setup time is less than 10us"

CALL following
CLS
NEXT i%

m$ = "ABORT (@5:8)"
CALL WRITE
CLS
GOTO beginning
```

```
'=============================================================
'            Adjusting analog outputs
'=============================================================

label2:

CLS
m$ = "OUTP ON,(@5:8)"
CALL WRITE

FOR i% = 1 TO 4
LOCATE 2, 20: PRINT "ADJUSTING ANALOG OUTPUTS"

LOCATE 11, 5: PRINT "Connect the DVM on output "; STR$(i% + 4)
LOCATE 12, 5: PRINT "Adjust PR"; STR$((i% * 2) - 1); " to obtain - 10.000V
   ±0.3mV"

m$ = "VOLT -10,(@5:8)"
CALL WRITE

CALL following

LOCATE 12, 5: PRINT "                                    "
LOCATE 12, 5: PRINT "Adjust PR"; STR$(i% * 2); " to obtain 9.9997V ± 0.3mV"

m$ = "VOLT 10,(@5:8)"
CALL WRITE
CALL following

LOCATE 12, 5: PRINT "                                      "
LOCATE 12, 5: PRINT "Make sure the offset is 0.000V ± 0.3mV"

m$ = "VOLT 0,(@5:8)"
CALL WRITE
CALL following
CLS
NEXT i%

m$ = "TEST? (@5:8)"
CALL WRITE
CALL READ

GOTO beginning
```

```
'========================================================
'              Adjust input filters
'========================================================

label3:
CLS
m$ = "OUTP ON,(@1:4)"
CALL WRITE

FOR i% = 1 TO 4

LOCATE 2, 20: PRINT "ADJUST INPUT FILTERS"

LOCATE 12, 5: PRINT "Connect the scope on the jumper J"; STR$(i% + 1); ",
   ground on TP13"
CALL following
LOCATE 13, 5: PRINT "Apply a square wave of 25kHz, ±10V to input";
   STR$(i%)
CALL following
LOCATE 14, 5: PRINT "Adjust PC"; STR$(i%); " and PC"; STR$(i% + 4); " for a
   slew rate of 3.5V/us"

CALL following
CLS
NEXT i%

GOTO beginning

'========================================================
'           Adjusting analog inputs
'========================================================

label4:

m$ = "OUTP ON,(@1:4)"
CALL WRITE
```

```
FOR i% = 1 TO 4
CLS

LOCATE 2, 20: PRINT "ADJUSTING ANALOG INPUTS"

LOCATE 5, 5: PRINT "1) Apply -9.000V to input"; STR$(i%);
LOCATE 6, 5: PRINT "2) Adjust PR"; STR$((i% * 2) - 1); " to read - 9.0000V
   ±0.3mV"
LOCATE 7, 5: PRINT "3) Then apply 9.000V to input"; STR$(i%);
LOCATE 8, 5: PRINT "4) Adjust PR"; STR$(i% * 2); " to read 9.0000V ±0.3mV"
LOCATE 9, 5: PRINT "5) Return to Step 1 if necessary"
LOCATE 10, 5: PRINT "6) Short circuit input"; STR$(i%); " to read 0.0000V
   ±0.3mV"
COLOR 15
LOCATE 12, 5: PRINT "WARNING ! these two adjustments interact"
COLOR 7

LOCATE 23, 33: PRINT "More..."

vnoise# = 0
DO

CALL measure
LOCATE 14, 5: PRINT "Min = "; USING "###.##### V"; mmin#
COLOR 15
LOCATE 18, 40: PRINT "Measure = "; USING "###.##### V"; m#
COLOR 7
LOCATE 16, 5: PRINT "Max = "; USING "###.##### V"; mmax#

IF ABS(mmax# - mmin#) * 1000 > vnoise# THEN vnoise# = ABS(mmax# -
   mmin#) * 1000
IF vnoise# > 10 THEN vnoise# = 0
COLOR 15
LOCATE 18, 5: PRINT "Vnoise = "; USING "####.## mV"; vnoise#
COLOR 7
v$ = UCASE$(INKEY$)

LOOP WHILE v$ = ""

CLS
NEXT i%

m$ = "TEST? (@1:4)"
CALL WRITE
CALL READ
GOTO beginning
END
```

```
'===========================================================
'                  Sub-routine for Measuring
'===========================================================

SUB measure:
m# = 0

m$ = "MEAS:VOLT? (@" + LTRIM$(STR$(i%)) + ")"
CALL WRITE
CALL READ

      mmin# = VAL(r$)
      mmax# = VAL(r$)

FOR j% = 1 TO 20
m$ = "MEAS:VOLT? (@" + LTRIM$(STR$(i%)) + ")"
CALL WRITE
CALL READ
mm# = VAL(r$)

IF mm# < mmin# THEN mmin# = mm#
IF mm# > mmax# THEN mmax# = mm#

m# = m# + mm#
NEXT j%
m# = m# / 20
EXIT SUB

'===========================================================
'                 Sub-routine for writing IEEE messages
'===========================================================

SUB WRITE:
' m$ contains the character string sent to slot 0

EXIT SUB

'===========================================================
'                  Subroutine for reading IEEE messages
'===========================================================

SUB READ:
      r$ = SPACE$(80)

      r$ = LTRIM$(RTRIM$(BUFFER$))
EXIT SUB
```

```
'===========================================================
'          Keyboard input routine
'===========================================================
SUB choice (v%)

DO
a$ = UCASE$(INKEY$)
IF a$ = " " THEN v% = 20: EXIT SUB
IF a$ <> "" THEN
   v% = ASC(a$)

   SELECT CASE v%
      CASE 49 TO 52
         EXIT SUB
      CASE 83
         EXIT SUB
      CASE ELSE
         v% = 0
         BEEP
      EXIT SUB
   END SELECT
ENDIF
```