MS2665C/67C/68C
Spectrum Analyzer
Operation Manual
Vol. 3
(Programming)

Eighth Edition

Read this manual before using the equipment.
Keep this manual with the equipment.

ANRITSU CORPORATION

Document No.: M-W1335AE-8.0

Safety Symbols

To prevent the risk of personal injury or loss related to equipment malfunction, Anritsu Corporation uses the following safety symbols to indicate safety-related information. Insure that you clearly understand the meanings of the symbols BEFORE using the equipment. Some or all of the following five symbols may not be used on all Anritsu equipment. In addition, there may be other labels attached to products which are not shown in the diagrams in this manual.

Symbols used in manual

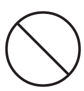
DANGER ⚠ This indicates a very dangerous procedure that could result in serious injury or death if not performed properly.

WARNING ⚠ This indicates a hazardous procedure that could result in serious injury or death if not performed properly.

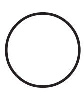
CAUTION ⚠ This indicates a hazardous procedure or danger that could result in light-to-severe injury, or loss related to equipment malfunction, if proper precautions are not taken.

Safety Symbols Used on Equipment and in Manual

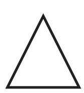
The following safety symbols are used inside or on the equipment near operation locations to provide information about safety items and operation precautions. Insure that you clearly understand the meanings of the symbols and take the necessary precautions BEFORE using the equipment.



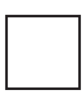
This indicates a prohibited operation. The prohibited operation is indicated symbolically in or near the barred circle.



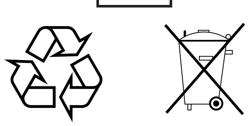
This indicates an obligatory safety precaution. The obligatory operation is indicated symbolically in or near the circle.



This indicates warning or caution. The contents are indicated symbolically in or near the triangle.



This indicates a note. The contents are described in the box.



These indicate that the marked part should be recycled.

MS2665C/67C/68C
Spectrum Analyzer
Operation Manual Vol. 3 (Programming)

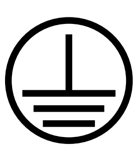
28 November 1997 (First Edition)
28 October 2004 (Eighth Edition)

For Safety

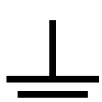
WARNING ⚠

1. ALWAYS refer to the operation manual when working near locations at which the alert mark shown on the left is attached. If the operation, etc., is performed without heeding the advice in the operation manual, there is a risk of personal injury. In addition, the equipment performance may be reduced.
   Moreover, this alert mark is sometimes used with other marks and descriptions indicating other dangers.

2. Measurement Categories
   This instrument is designed for Measurement category I (CAT I). Don't use this instrument at the locations of measurement categories from CAT II to CAT IV.
   In order to secure the safety of the user making measurements, IEC 61010 clarifies the range of use of instruments by classifying the location of measurement into measurement categories from I to IV.
   The category outline is as follows:
   Measurement category I (CAT I):
   Secondary circuits of a device connected to an outlet via a power transformer etc.
   Measurement category II (CAT II):
   Primary circuits of a device with a power cord (portable tools, home appliance etc.) connected to an outlet.
   Measurement category III (CAT III):
   Primary circuits of a device (fixed equipment) to which power is directly supplied from the power distribution panel, and circuits from the distribution panel to outlets.
   Measurement category IV (CAT IV):
   All building service-line entrance circuits through the integrating watt-meter and primary circuit breaker (power distribution panel).



or



3. When supplying power to this equipment, connect the accessory 3-pin power cord to a grounded outlet. If a grounded outlet is not available, before supplying power to the equipment, use a conversion adapter and ground the green wire, or connect the frame ground on the rear panel of the equipment to ground. If power is supplied without grounding the equipment, there is a risk of receiving a severe or fatal electric shock.

For Safety

WARNING ⚠

Repair

WARNING ⚠

4. This equipment cannot be repaired by the user. DO NOT attempt to open the cabinet or to disassemble internal parts. Only Anritsu-trained service personnel or staff from your sales representative with a knowledge of electrical fire and shock hazards should service this equipment. There are high-voltage parts in this equipment presenting a risk of severe injury or fatal electric shock to untrained personnel. In addition, there is a risk of damage to precision parts.

Falling Over

5. This equipment should be used in the correct position. If the cabinet is turned on its side, etc., it will be unstable and may be damaged if it falls over as a result of receiving a slight mechanical shock.
   And also DO NOT use this equipment in the position where the power switch operation is difficult.

Battery Fluid

6. DO NOT short the battery terminals and never attempt to disassemble it or dispose of it in a fire. If the battery is damaged by any of these actions, the battery fluid may leak.
   This fluid is poisonous.
   DO NOT touch it, ingest it, or get in your eyes. If it is accidentally ingested, spit it out immediately, rinse your mouth with water and seek medical help. If it enters your eyes accidentally, do not rub your eyes, irrigate them with clean running water and seek medical help. If the liquid gets on your skin or clothes, wash it off carefully and thoroughly.

LCD

7. This instrument uses a Liquid Crystal Display (LCD); DO NOT subject the instrument to excessive force or drop it. If the LCD is subjected to strong mechanical shock, it may break and liquid may leak.
   This liquid is very caustic and poisonous.
   DO NOT touch it, ingest it, or get in your eyes. If it is ingested accidentally, spit it out immediately, rinse your mouth with water and seek medical help. If it enters your eyes accidentally, do not rub your eyes, irrigate them with clean running water and seek medical help. If the liquid gets on your skin or clothes, wash it off carefully and thoroughly.

For Safety

CAUTION ⚠

Replacing Fuse

CAUTION ⚠

1. Before Replacing the fuses, ALWAYS remove the power cord from the poweroutlet and replace the blown fuses. ALWAYS use new fuses of the type and rating specified on the fuse marking on the rear panel of the cabinet.

   T5A indicates a time-lag fuse.

   There is risk of receiving a fatal electric shock if the fuses are replaced with the power cord connected.

Cleaning

2. Keep the power supply and cooling fan free of dust.
   - Clean the power inlet regularly. If dust accumulates around the power pins, there is a risk of fire.
   - Keep the cooling fan clean so that the ventilation holes are not obstructed. If the ventilation is obstructed, the cabinet may overheat and catch fire.

Check Terminal

3. - Maximum DC voltage ratings:
     RF Input DC 0 V
   - Maximum AC power ratings:
     RF Input +30 dBm
   - NEVER input a >+30 dBm and >DC 0 V power to RF Input.
   - Excessive power may damage the internal circuits.

For Safety

CAUTION ⚠

Replacing Memory Back-up Battery

This equipment uses a Poly-carbomonofluoride lithium battery to back-up the memory. This battery must be replaced by a service engineer when it has reached the end of its useful life; contact the Anritsu sales section or your nearest representative.

Note: The battery used in this equipment has a maximum useful life of 7 years. It should be replaced before this period has elapsed.

External Storage Media

This equipment stores data and programs using Plug-in Memory card (MC).
Data and programs may be lost due to improper use or failure.
ANRITSU therefore recommends that you backup the memory.

<u>ANRITSU CANNOT COMPENSATE FOR ANY MEMORY LOSS.</u>

Please pay careful attention to the following points.

- Do not remove the IC card from equipment being accessed.
- Isolate the card from static electricity.
- The backup battery in the SRAM memory card has a limited life; replace the battery periodically.

Disposing of The Product

This equipment uses chemical compound semiconductor including arsenide.
At the end of its life, the equipment should be recycled or disposed properly according to the local disposal regulations.

Equipment Certificate

Anritsu Corporation certifies that this equipment was tested before shipment using calibrated measuring instruments with direct traceability to public testing organizations recognized by national research laboratories including the National Institute of Advanced Industrial Science and Technology, and the National Institute of Information and Communications Technology, and was found to meet the published specifications.

Anritsu Warranty

Anritsu Corporation will repair this equipment free-of-charge if a malfunction occurs within 1 year after shipment due to a manufacturing fault, provided that this warranty is rendered void under any or all of the following conditions.

- The fault is outside the scope of the warranty conditions described in the operation manual.
- The fault is due to mishandling, misuse, or unauthorized modification or repair of the equipment by the customer.
- The fault is due to severe usage clearly exceeding normal usage.
- The fault is due to improper or insufficient maintenance by the customer.
- The fault is due to natural disaster including fire, flooding, earthquake, etc.
- The fault is due to use of non-specified peripheral equipment, peripheral parts, consumables, etc.
- The fault is due to use of a non-specified power supply or in a non-specified installation location.

In addition, this warranty is valid only for the original equipment purchaser. It is not transferable if the equipment is resold.

Anritsu Corporation will not accept liability for equipment faults due to unforeseen and unusual circumstances, nor for faults due to mishandling by the customer.

Anritsu Corporation Contact

If this equipment develops a fault, contact Anritsu Service and Sales offices at the address at the end of paper-edition manual or the separate file of CD-edition manual.

Front Panel Power Switch

To prevent malfunction caused by accidental touching, the front power switch of this equipment turns on the power if it is pressed continuously for about one second in the standby state. If the switch is pressed continuously for one second in the power-on state, the equipment enters the standby state.

In the power-on state, if the power plug is removed from the outlet, then reinserted into it, the power will not be turned on. Also, if the lines is disconnected due to momentary power supply interruption or power failure, the power will not be turned on (enters the standby state) even if the line is recovered.

This is because this equipment enters the standby state and prevents incorrect data from being acquired when the line has to be disconnected and reconnected.

For example, if the sweep time is 1,000 seconds and data acquisition requires a long time, momentary power supply interruption (power failure) might occur during measurement and the line could be recovered automatically to power-on. In such a case, the equipment may mistake incorrect data for correct data without recognizing the momentary power supply interruption.

If this equipment enters the standby state due to momentary power supply interruption or power failure, check the state of the measuring system and press the front power switch to restore power to this equipment.

Further, if this equipment is built into a system and the system power has to be disconnected then reconnected, the power for this equipment must also be restored by pressing the front power switch.

Consequently, if this equipment is built into remote monitoring systems that use MODEMs, the standby function of this equipment must be modified.

ABOUT DETECTION MODE

This instrument is a spectrum analyer which uses a digital storage system. The spectrum analyzer makes level measurements in frequency steps obtained by dividing the frequency span by the number of measurement data points (501). This method of measurement cannot detect the signal peak level if the spectrum of a received signal is narrower than these frequency steps.
To resolve this problem, this instrument usually operates in positive peak detection mode and normal detection mode. In the positive peak detection mode, the highest level within the frequency range between the sample points can be held and traced. In the normal detection mode, both the positive peak and the negative peak can be traced.
Positive peak detection mode should be used for almost all measurements including normal signal level measurement, pulsed noise analysis, and others. <u>It is impossible to measure the signal level accurately in sample detection mode or in negative peak detection mode.</u>
Use of sample detection mode is restricted to random noise measurement, occupied frequency bandwidth measurement for analog communication systems, and adjacent-channel leakage power measurement, etc.

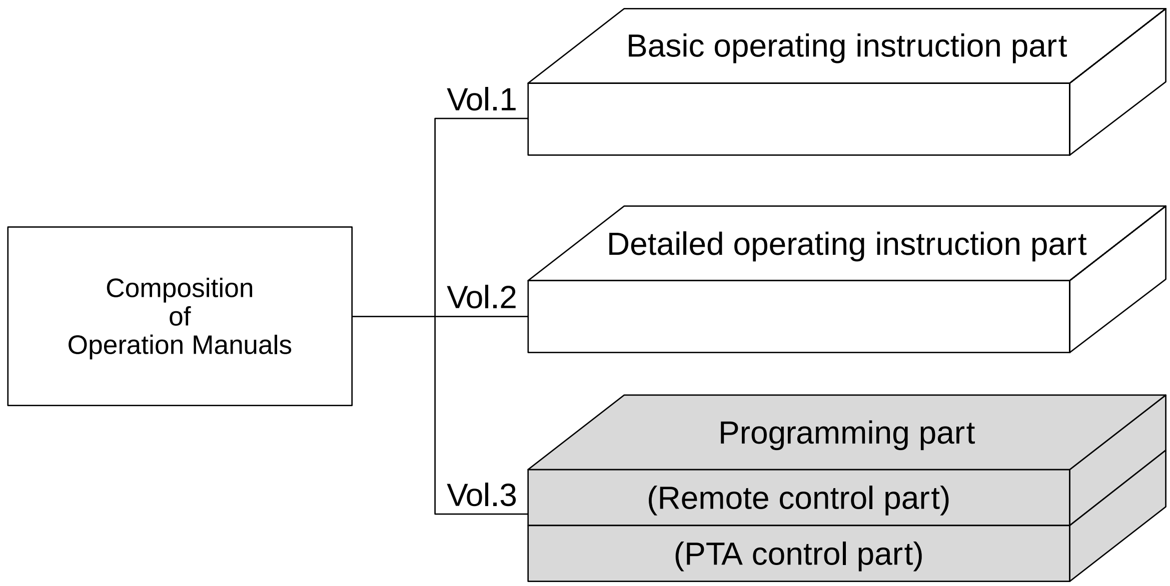
Measurement	item
• Normal signal	POS PEAK
• Random noise	SAMPLE
• Pulsed noise	NORMAL (POSI-NEG)
• Occupied frequency bandwidth, adjacent-channel leakage power (for analog communication systems)	SAMPLE
• Occupied frequency bandwidth, adjacent-channel leakage power (for digital communication systems)	POS PEAK or SAMPLE

When a detection mode is specified as one of the measurement methods, make the measurement in the specified detection mode.

ABOUT THIS MANUAL

(1) Composition of MS2665C/67C/68C spectrum analyzer Operation Manuals

The MS2665C/67C/68C Spectrum Analyzer operation manuals of the standard type are composed of the following three documents. Use them properly according to the usage purpose.



Basic operating instruction part:	Basic Operating Instructions: Provides information on the MS2665C/67C/68C outline, preparation before use, panel description, basic operation, soft-key menu and performance tests.
Detailed operating instruction part:	Detailed Operating Instructions: Provides information on the detailed panel operating instructions on the spectrum analyzer that expand on the basic operation and soft-key menu in the Basic Operating Instruction Part.
Programming part:	Composed of the Remote Control Part and PTA Control Part. The Remote Control Part provides information on RS-232C remote control GPIB remote control and sample programs, while the PTA Control Part describes about PTA operation and PTL commands.

TABLE OF CONTENTS

MS2665C/67C/68C
Spectrum Analyzer
Operation Manual
Programming
(Remote Control)

TABLE OF CONTENTS

SECTION 1

GENERAL

This section outlines the remote control and gives examples of system upgrades.

TABLE OF CONTENTS

SECTION 1 GENERAL

General

The MS2665C/67C/68C Spectrum Analyzer, when combined with an external controller (host computer, personal computer, etc.), can automate your measurement system. For this purpose, the spectrum analyzer is equipped with an RS-232C interface port, GP-IB interface bus (IEEE std 488.2-1987).

Remote control functions

The remote control functions of the MS2665C/67C/68C are used to do the following:

(1) Control all functions except a few like the power switch and [LOCAL] key

(2) Read all parameter settings.

(3) Set the RS-232C interface settings from the panel

(4) Set the GP-IB address from the panel

(5) Select the interface port application from the panel

(6) Configure the automatic measurement system when the spectrum analyzer is combined with a personal computer and other measuring instruments.

Interface port selection functions

The MS2665C/67C/68C Spectrum Analyzer has a standard RS-232C interface, and an optional GP-IB interface bus and parallel (Centro) interface (option 10). Use the panel to select the interface port to be used to connect external devices as shown below.

Port for the external controller: Select RS-232C or GP-IB.

Port for the printer or plotter: Select RS-232C or GP-IB or Centro.

Port for the external device controlled from the PTA: Select RS-232C or GP-IB or Centro.

Each interface can connect only one device.

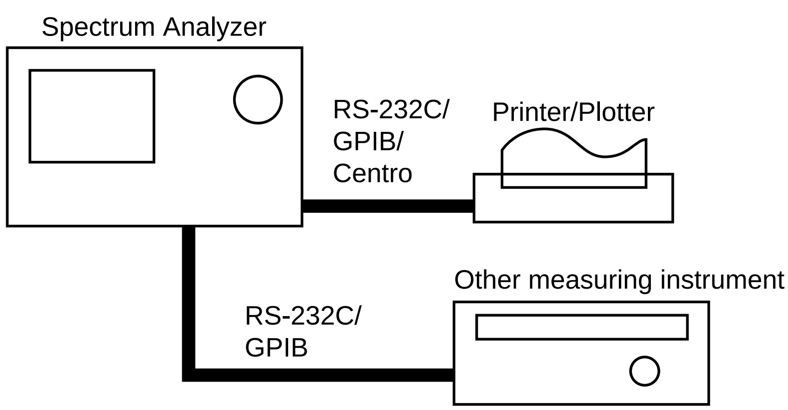
Examples of system upgrades using RS-232C and GP-IB

(1) Stand-alone type 1

Waveforms measured with the MS2665C/67C/68C are output to the printer and plotter.

(2) Stand-alone type 2

Other measuring instruments are controlled from the PTA. The printer, plotter, and external device controlled from the PTA must be connected using different interfaces.

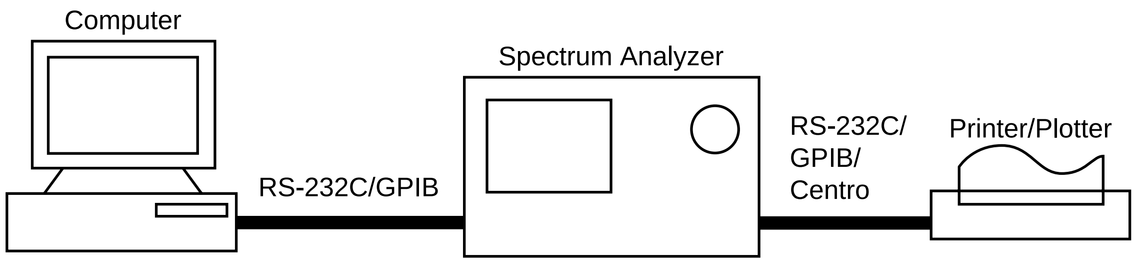


(3) Control by the host computer (1)

The spectrum analyzer is controlled automatically or remotely from the computer.

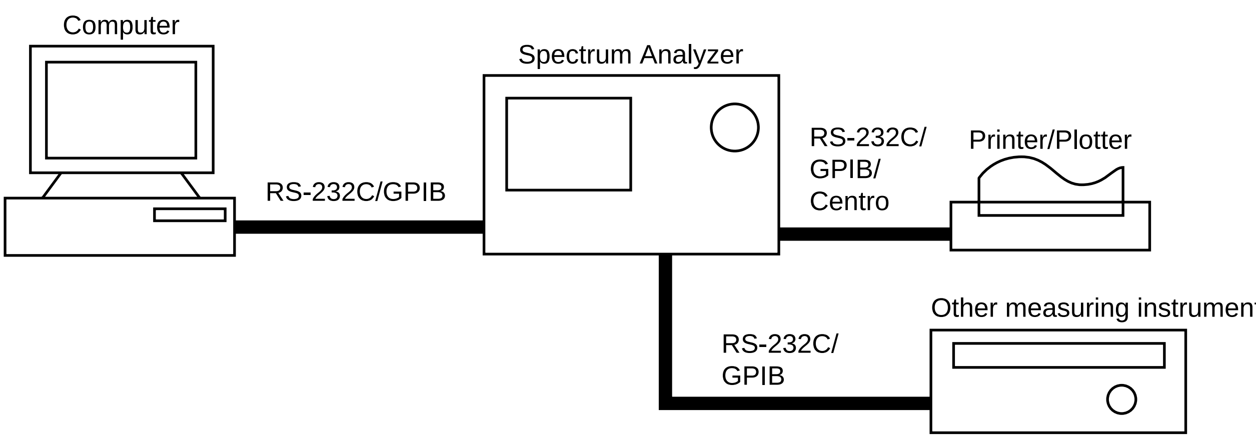
(4) Control by the host computer (2)

The waveforms measured by controlling spectrum analyzer automatically or remotely are output to the printer and plotter. The external controller, printer, and plotter must be connected using different interfaces.



(5) Control by the host computer (3)

The waveforms measured by controlling the spectrum analyzer automatically or remotely are output to the printer and plotter. PTA programs are executed from the computer. The printer, plotter, and external device controlled from the PTA must be connected using different interfaces.



Specifications of RS-232C

The table below lists the specifications of the RS-232C provided as standard in the MS2665C/67C/68C.

Item	Specification
Function	Outputs printing data to the printer and plotter. Control from the external controller (except for power-ON/OFF)
Communication system	Asynchronous (start-stop synchronous system), half-duplex
Communication control system	X-ON/OFF control
Baud rate	
Data bits	7 or 8 bits
Parity	Odd number (ODD), even number (EVEN), none (NON)
Start bit	1 bit
Stop bit (bits)	1 or 2 bits
Connector	D-sub 9-pin, female

Specifications of GP-IB

The table below lists the specifications of the GP-IB provided for the MS2665C/67C/68C.

Item	Specification and supplementary explanation
Function	Conforms to IEEE488.2 The spectrum analyzer is controlled from the external controller (except for power-on/off). The spectrum analyzer is used as a controller for an external device (printer or plotter).
Interface function (*1)	SH1: All source handshake functions are provided. Synchronizes the timing of data transmission. AH1: All acceptor handshake functions are provided. Synchronizes the timing of data reception. T6: The basic talker functions and serial poll function are provided. The talk only function is not provided. The talker can be canceled by MLA. L4: The basic listener functions are provided. The listenonly function is not provided. The listener can be canceled by MTA. SR1: All service request and status byte functions are provided. RL1: All remote/local functions are provided. The local lockout function is provided. PP0: The parallel poll functions are not provided. DC1: All device clear functions are provided. DT1: Device trigger functions are provided. C1: System controller functions are provided. C2: IEC is transmitted. C3: The REN transmission function is provided. C4: Responses to SRQ are returned. C28: Interface messages are transmitted. E2: Output is tri-state.

*1 For details of the interface functions, see the GP-IB Basic Guide sold separately.

SECTION 2

CONNECTING DEVICE

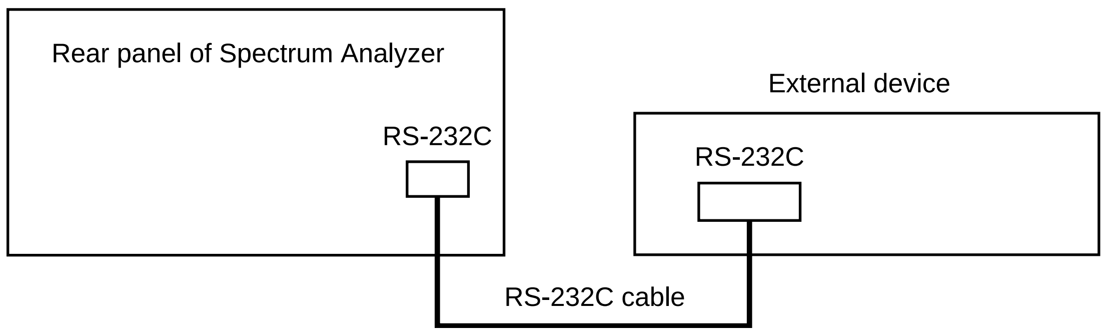
This section describes how to connect external devices such as the host computer, personal computer, printer, and plotter with RS-232C and GP-IB cables. This section also describes how to setup the interfaces of the spectrum analyzer.

TABLE OF CONTENTS

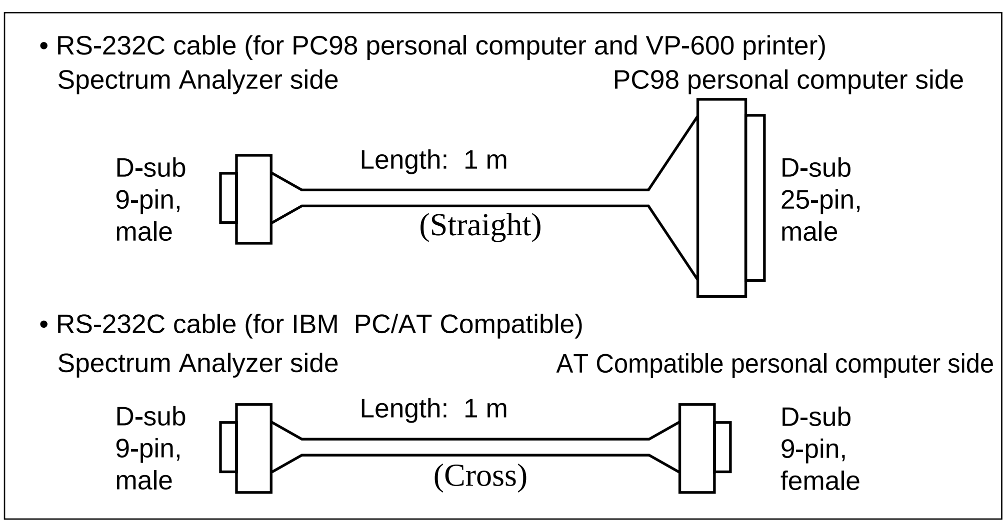
SECTION 2
CONNECTING DEVICE

Connecting an external device with an RS-232C cable

Connect the RS-232C connector (D-sub 9-pin, female) on the rear panel of the spectrum analyzer to the RS-232C connector of the external device with an RS-232C cable.



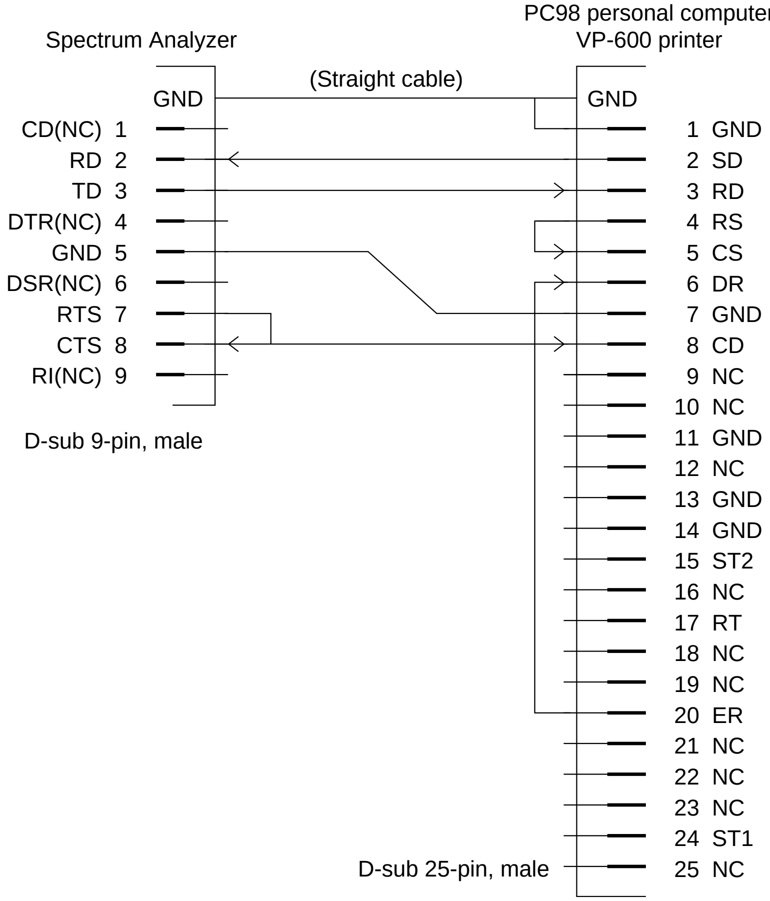
Notes: *RS-232C connectors with 9 pins and 25 pins are available. When purchasing the RS-232C cable, check the number of pins on the RS-232C connector of the external device. Also, the following RS232C cables are provided as peripheral parts of the spectrum analyzer.*



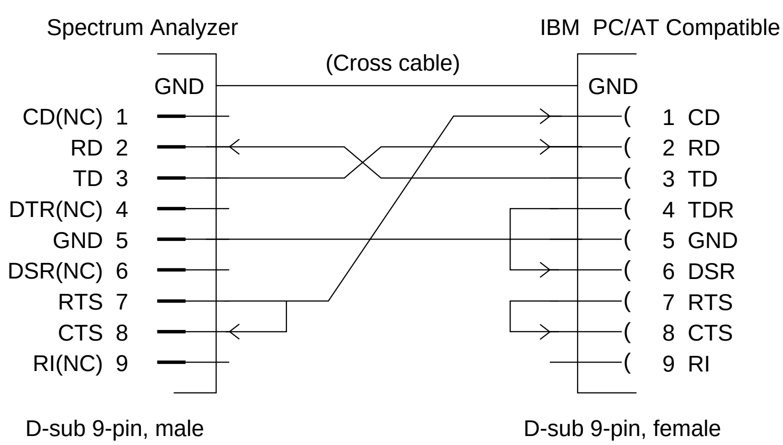
Connection diagram of RS-232C interface signals

The diagram below shows the RS-232C interface signal connections between the spectrum analyzer and devices such as a personal computer or printer.

• Connection with PC98 personal computer or VP-600 printer



• Connection with IBM PC/AT Compatible personal computer



Setting the connection port interfaces

Set the interfaces between connection ports of the spectrum analyzer and external devices such as a personal computer, printer, or plotter.

Shift → . (Interface)

RS-232C Setup *	Set the Baud rate (1200 to 9600), parity (Off, Even, or Odd), data bit (bit 7 or 8), and stop bit (bit 1 or 2) of the RS-232C.
Connect to Controller GPIB	Select None, GP-IB, or RS-232C as the interface of the connection port for the external controller. The selection is changed each time this key is pressed.
Connect to Prt/Plt RS-232C	Select None, GP-IB, RS-232C, or Centro for the interface of the connection port for the printer or plotter. The selection is changed each time this key is pressed.
Connect to Peripheral None	Select None, GP-IB, RS-232C, or Centro for the interface of the connection port for the peripheral equipment controlled by the PTA. The selection is changed each time this key is pressed.

In the above example, the GP-IB interface is selected for the connection port for the external controller, and the RS-232C interface is selected for the connection port for the printer or plotter.

Setting the RS-232C interface conditions

Set the RS-232C interface conditions of this equipment to those of the external device to be connected.

Shift → Interface [.] → RS-232C Setup *

→ Baud Rate 9600bps	Select a Baud rate of 1200, 2400, 4800, or 9600 bits per second. The selection is changed each time this key is pressed.
→ Parity Even	Press this key to select the parity (Off, Even, or Odd).
→ Data Bits 8bits	Press this key to select the number of data bits (7 or 8 bits).
→ Stop bit 1bit	Press this key to select the number of stop bits (1 or 2 bits).
→ return	

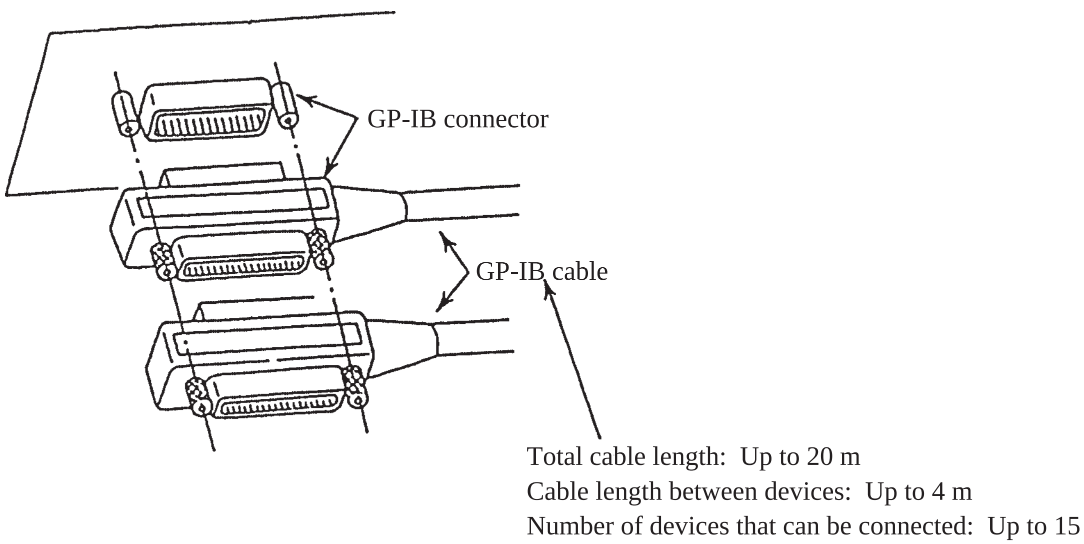
For how to set the RS-232C interface of an external device, see the operation manual of the external device.

Connecting a device with a GP-IB cable

Connect the GP-IB connector on the rear panel of this equipment to the GP-IB connector of an external device with a GP-IB cable.

Note: Be sure to connect the GP-IB cable before turning the equipment power on.

Up to 15 devices, including the controller, can be connected to one system. Connect devices as shown below.



Setting the GP-IB address

Set the GPIB address of this equipment as follows.

Shift → Interface [.]

→ GP-IB
My Address 1

Use the 10-key pad to enter the GP-IB address of this equipment.
The initial value is 1.

For how to set the GPIB address of an external device, see the operation manual of the external device.

SECTION 3

DEVICE MESSAGE FORMAT

This section describes the format of the device messages transmitted on the bus between a controller (host computer) and device (MS2665C/67C/68C) via the RS-232C or GP-IB system.

TABLE OF CONTENTS

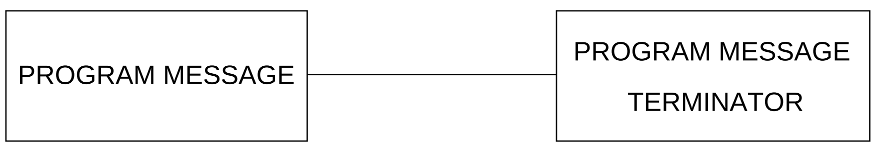
SECTION 3
DEVICE MESSAGE FORMAT

General description

The device messages are data messages transmitted between the controller and devices, program messages transferred from the controller to this instrument (device), and response messages input from this instrument (device) to the controller. There are also two types of program commands and program queries in the program message. The program command is used to set this instrument's parameters and to instruct it to execute processing. The program query is used to query the values of parameters and measured results.

Program message format

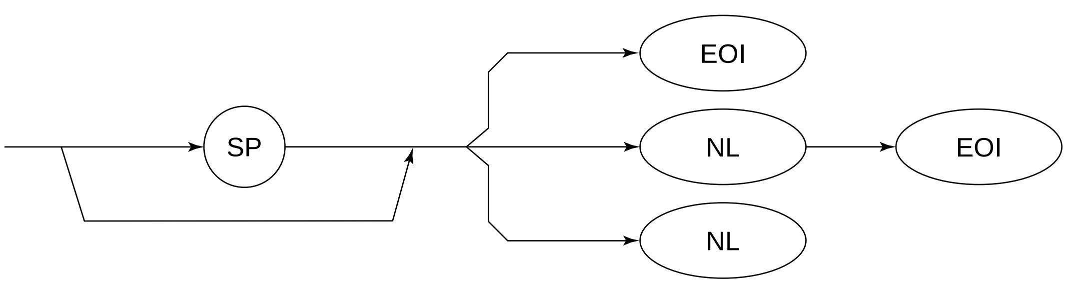
To transfer a program message from the controller program to this instrument using the WRITE statement, the program message formats are defined as follows.




```
WRITE  #1,"CF :1GHZ"
```

PROGRAM MESSAGE: When the program message is transmitted from the controller to this instrument, the specified terminator is attached to the end of the program message to terminate its transmission.

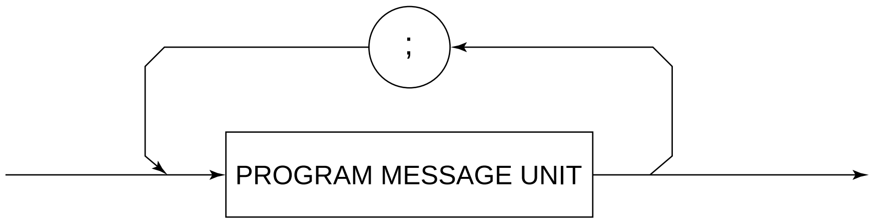
(1) PROGRAM MESSAGE TERMINATOR



NL: Called New line or LF (Line Feed)

Carriage Return (CR) is ignored and is not processed as a terminator.

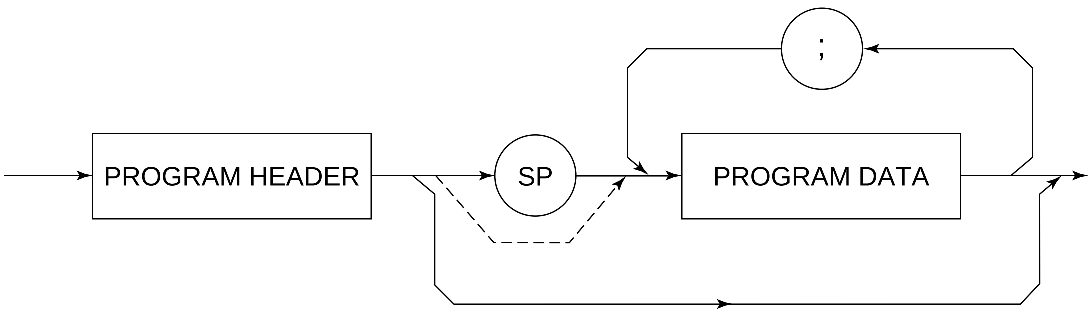
(2) PROGRAM MESSAGE



Multiple program message units can be output sequentially by separating them with a semicolon.

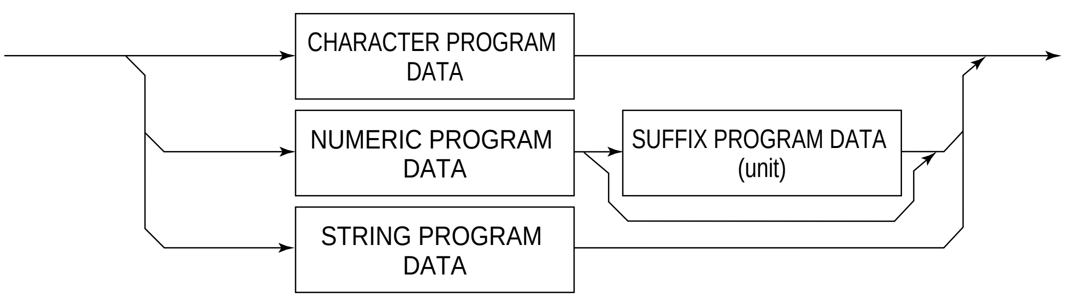
<Example> `WRITE #1;"CF 1GHZ;SP 5ØØKHZ`

(3) PROGRAM MESSAGE UNIT



- The program header of an IEEE488.2 common command always begins with an asterisk.
- For numeric program data, the SP between the header and data can be omitted.
- The program header of a program query always ends with a question mark.

(4) PROGRAM DATA



(5) CHARACTER PROGRAM DATA

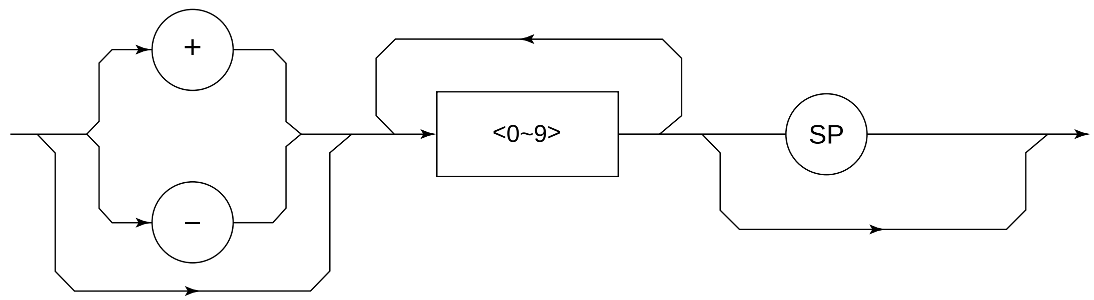
Character program data is specific character string data consisting of the uppercase alphabetic characters from A to Z, lowercase alphabetic characters from a to z, numbers 0 to 9, and underline (_).

<Example> `WRITE #1;"ST AUTO"`.............. Sets Sweep Time to AUTO.

(6) NUMERIC PROGRAM DATA

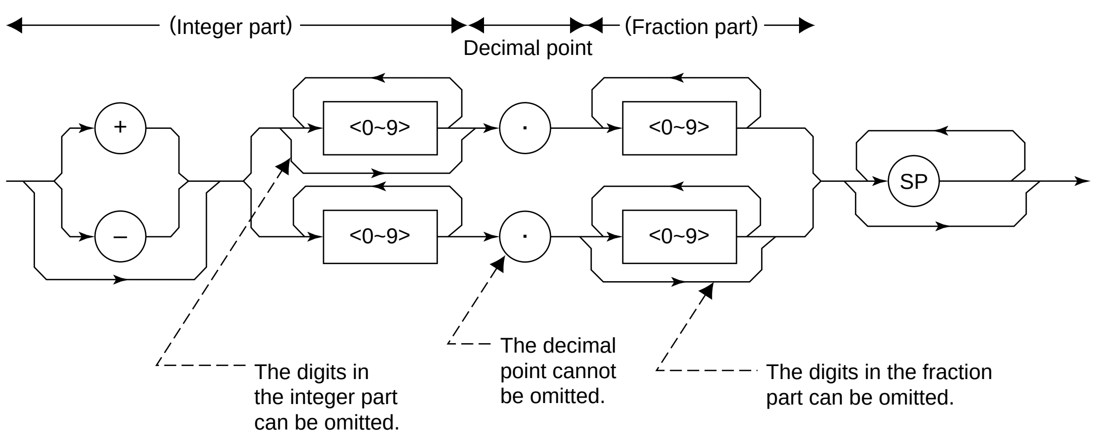
Numeric program data has two types of formats: integer format (NR1) and fixed-point format (NR2).

< Integer format (NR1) >



- Zeros can be inserted at the beginning → 005, +000045
- There must be no spaces between a + or – sign and a number → +5, +△ 5 (×)
- Spaces can be inserted after a number → +5△△△
- The + sign is optional → +5, 5
- Commas cannot be used to separate digits → 1,234,567 (×)

<Fixed-point format (NR2)>



- The numeric expression of the integer format applies to the integer part.
- There must be no spaces between numbers and the decimal point → +753△.123 (×)
- Spaces can be inserted after the digits in the fraction part → +753.123△△△
- A number need not be placed before the decimal point → .05
- A + or - sign can be placed before the decimal point → +.05, −.05
- A number can end with a decimal point → 12.

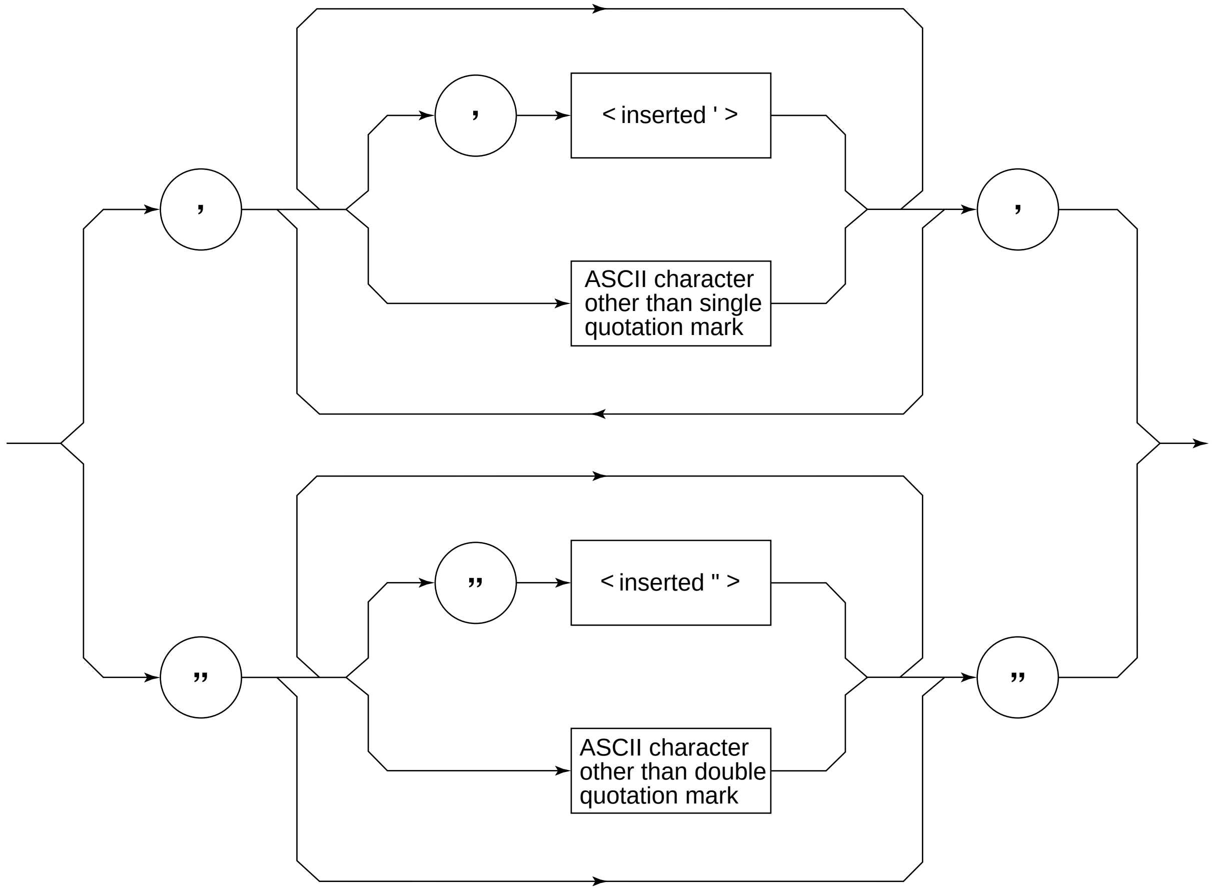
(7) SUFFIX PROGRAM DATA (unit)

The table below lists the suffixes used for the MS2665C/67C/68C.

Table of Suffix Codes

Classification	Unit	Suffix code
Frequency	GHz	GHZ, GZ
	MHz	MHZ, MZ
	kHz	KHZ, KZ
	Hz	HZ
	Default	HZ
Time	s	S
	ms	MS
	μs	US
	Default	MS
Level (dB system)	dB	DB
	dBm	DBM,DM
	dBμV	DBUV
	dBmV	DBMV
	dBμV(emf)	DBUVE
	Default	Determined in conformance with the set scale unit
Level (V system)	V	V
	mV	MV
	μV	UV
	Default	UV
Level (W system)	W	W
	mW	MW
	μW	UW
	nW	NW
	pW	PW
	fW	FW
	Default	UW

(8) STRING PROGRAM DATA



- String program data must be enclosed with single quotation marks ('...').

```
WRITE #1:"TITLE'MS2665C'"
```

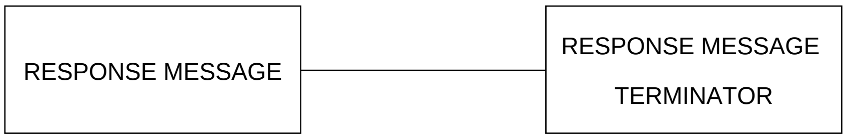
A single quotation mark used within a character string must be repeated as shown in the double quotation marks.

```
WRITE #1;"TITLE'MS2665C''NOISE MEAS'''"
```

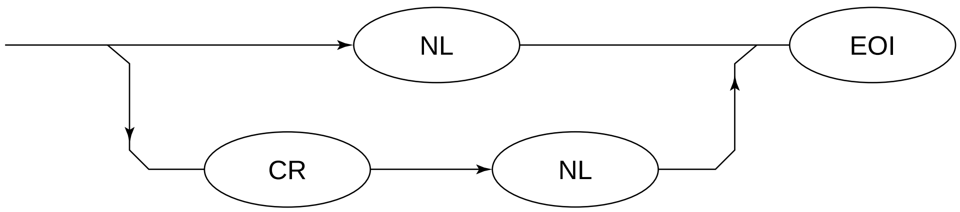
'NOISE MEAS' is set as the title.

Response message format

To transfer the response messages from this instrument to the controller using the READ statement, the response message formats are defined as follows.

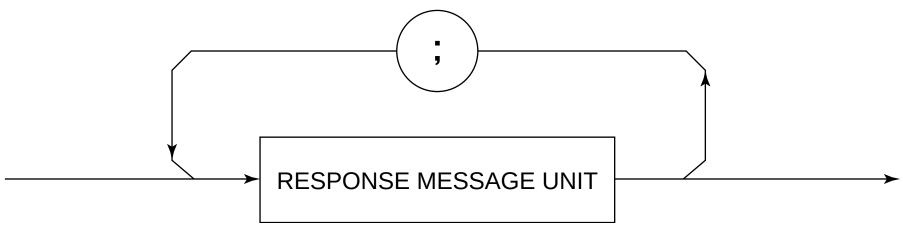


(1) RESPONSE MESSAGE TERMINATOR



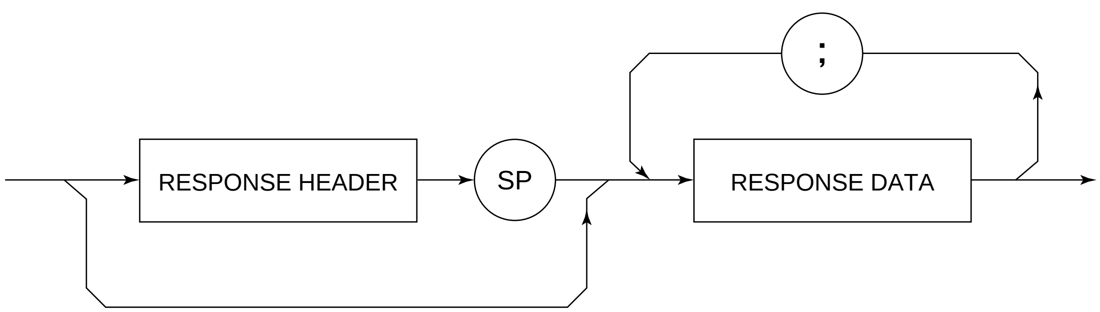
The response message terminator to be used depends on the TRM command specification.

(2) RESPONSE MESSAGE

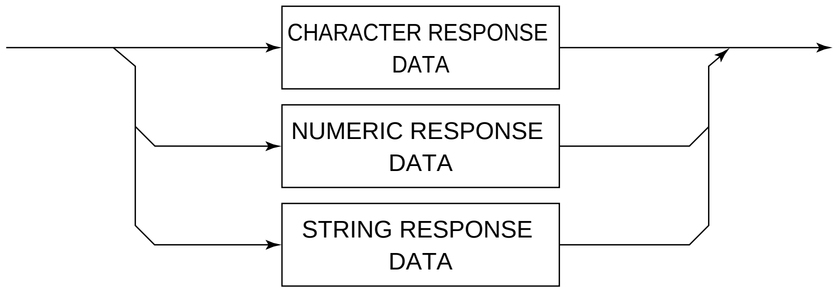


When a query is sent by the WRITE statement with one or more program queries, the response message also consists of one or more response message units.

(3) Usual RESPONSE MESSAGE UNIT



(4) RESPONSE DATA

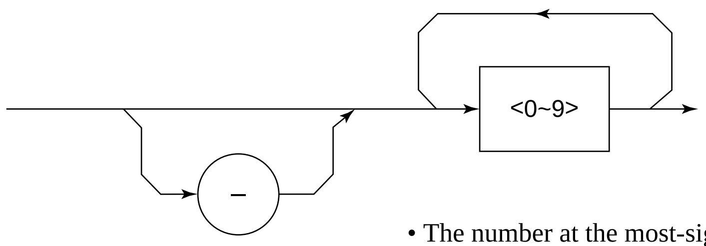


(5) CHARACTER RESPONSE DATA

Character response data is specific character string data consisting of the uppercase alphabetic characters from A to Z, lowercase alphabetic characters from a to z, numbers 0 to 9, and underline (_).

(6) NUMERIC RESPONSE DATA

< Integer format (NR1) >

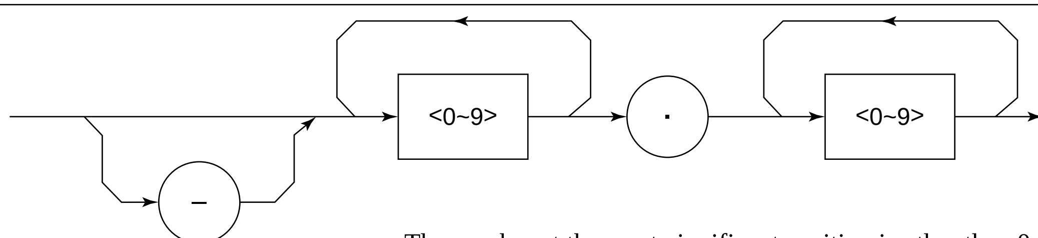


< Example >

```
123
–1234
```

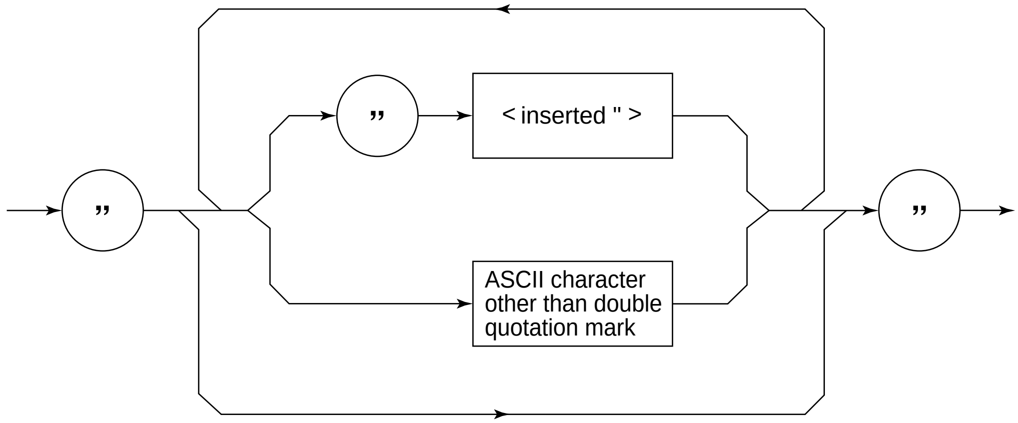
- The number at the most-significant position is other than 0.
- A + sign need not be placed before a positive number.

< Fixed-point format (NR2) >



- The number at the most-significant position is other than 0.
- A + sign need not be placed before a positive number.
- If there are zeros after the decimal point, the numeric data is transmitted in the integer format.

(7) CHARACTER RESPONSE DATA



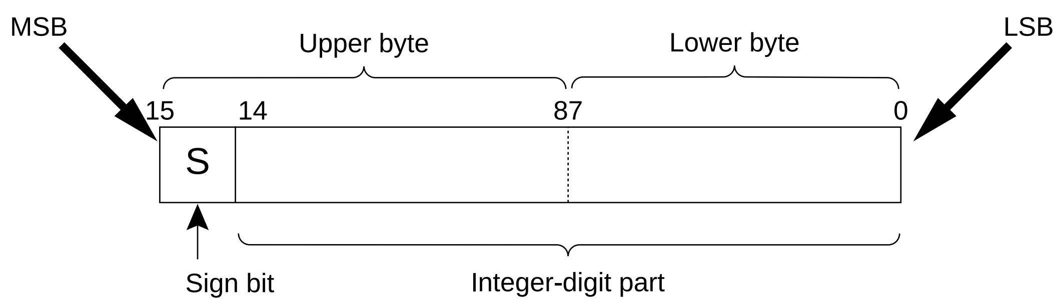
String response data is transmitted as an ASCII character enclosed with double quotation marks.

(8) Response message for input of waveform data using binary data

The waveform binary data is two-byte 65536 integer data from –32768 to 32767, as shown below; and sent in the sequence of upper byte and lower byte.

16-Bit Binary	With Sign	No Sign
1000000000000000	-32768	32768
1000000000000001	-32767	32769
1000000000000010	-32766	32770
1111111111111101	-3	65533
1111111111111110	-2	65534
1111111111111111	-1	65535
0000000000000000	0	0
0000000000000001	1	1
0000000000000010	2	2
0000000000000011	3	3
0111111111111101	32765	32765
0111111111111110	32766	32766
0111111111111111	32767	32767

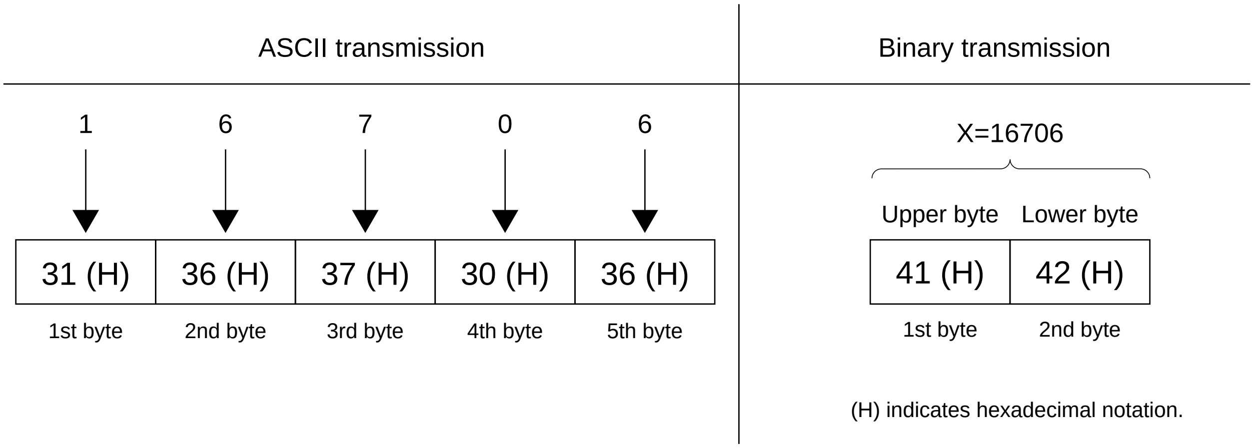
Sign bit: Bit 15(MSB) is used.
Integer-digit part: Bits 0 to 14 are used.
So, 16 bits(two bytes) are used.



Internal Representation of 2-byte Bynary Data

† When a negative number is stored in a numeric variable, the sign bit 1 is set in the MSB to indicate the negative value. The value is stored in a numeric variable in a 2's complement format.

For an example, to transmit an integer of 16706, the ASCII format is compared with the Binary format, below. The ASCII format requires 5 bytes. Whereas, the Binary format requires only 2 bytes, and does not need the data format transformation. So, The Binary format is used for a high-speed transmission.



$$16706\ (D) = 4 \times 16^3 + 1 \times 16^2 + 4 \times 16^1 + 2 \times 16^0$$

Binary notation

Upper byte | Lower byte

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X=16706	0	1	0	0	0	0	0	1	0	1	0	0	0	0	1	0

1st byte $\rightarrow$ 2nd byte (Transmission order)

bit	7	6	5	4	3	2	1	0	
1st byte	0	1	0	0	0	0	0	1	(Upper byte) = 41 (H)
2nd byte	0	1	0	0	0	0	1	0	(Lower byte) = 42 (H)

The waveform binary data has a number of bytes for

(Number of points to be specified) X 2 bytes + termination code.

Where, termination code is specified by the TRM command, and is LF(0D(H): 1 byte) or CR+LF(0A0D(H): 2 bytes).

SECTION 4

STATUS STRUCTURE

This section describes the device-status reporting and its data structure defined by the IEEE488.2 when the GP-IB interface bus is used. This section also describes the synchronization techniques between a controller and device.
These functions are used to control a device from an external controller using the GP-IB interface bus. Most of these functions can also be used to control a device from an external controller using the RS-232C interface

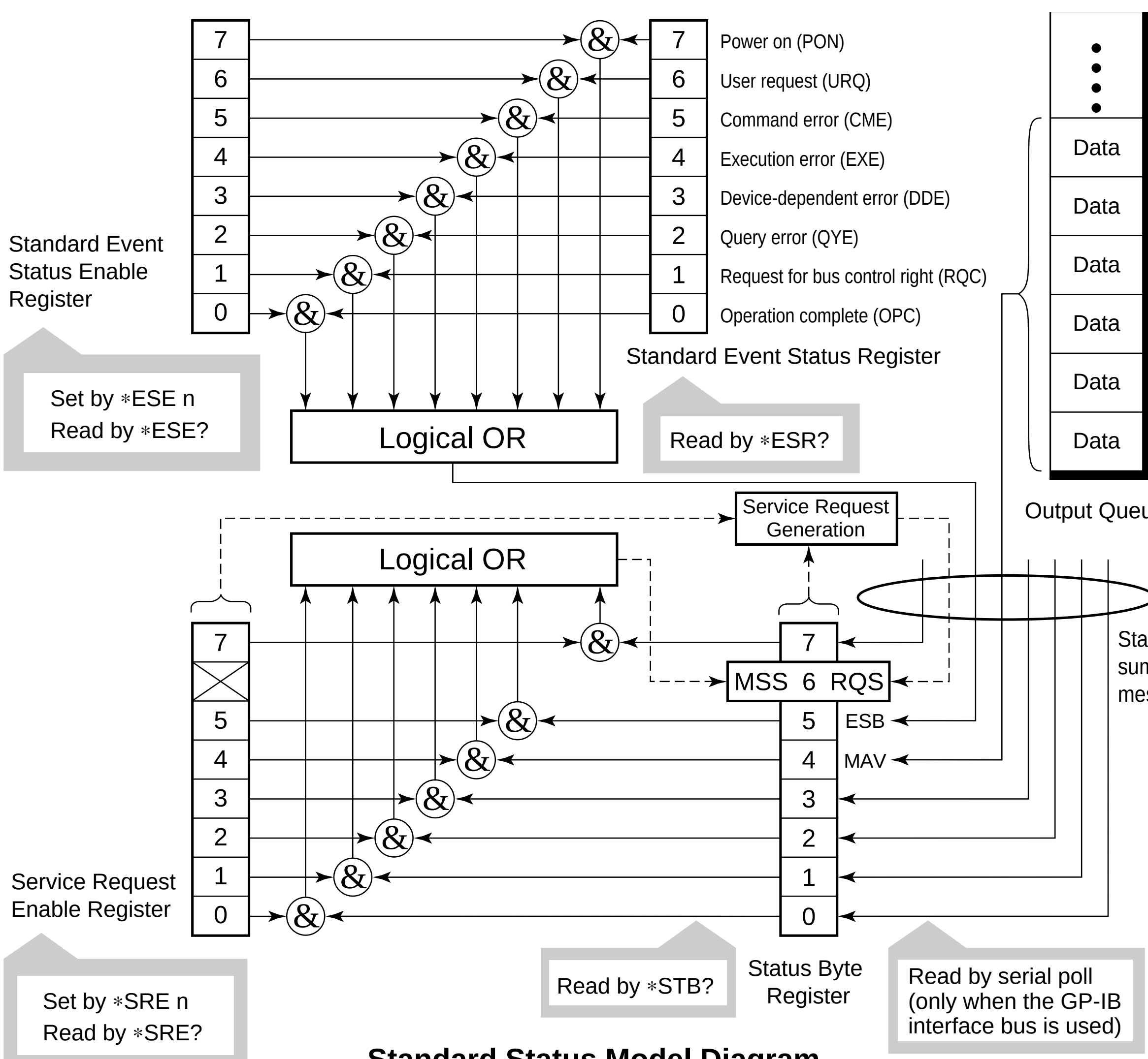
TABLE OF CONTENTS

SECTION 4
STATUS STRUCTURE

The Status Byte (STB) sent to the controller is based on the IEEE488.1 standard. The bits comprising the STB are called status summary messages because they represent a summary of the current data in registers and queues.

IEEE488.2 Standard Status Model

The diagram below shows the standard model for the status data structures stipulated in the IEEE488.2 standard.



Standard Status Model Diagram

In the status model, IEEE488.1 status bytes are used for the lowest grade status. This status byte is composed of seven summary message bits from the higher grade status structure. To create these summary message bits, the status data structure is composed of two types of register and queue models.

Register model	Queue model
The register model consists of two registers used for recording events and conditions encountered by a device. These two registers are the Event Status Register and Event Status Enable Register. When the results of the AND operation of both register contents are other than 0, the corresponding bit of the status bit becomes 1. In other cases, the corresponding bit becomes 0. When the result of their Logical OR is 1, the summary message bit also becomes 1. If the Logical OR result is 0, the summary message bit also becomes 0.	The queue in the queue model is used to sequentially record the waiting status values or information. If the queue is not empty, the queue structure summary message becomes 1. If the queue is empty, the message becomes 0.

In IEEE488.2, there are three standard models for the status data structure. Two are register models and one is a queue model based on the register model and queue model described above. The three standard models are:

1) Standard Event Status Register and Standard Event Status Enable Register
2) Status Byte Register and Service Request Enable Register Output Queue

Standard Event Status Register	Status Byte Register	Output Queue
The Standard Event Status Register has the same structure as the previously described register model. In this register, the bits for eight types of standard events encountered by a device are set as follows: 1) Power on 2) User request 3) Command error 4) Execution error 5) Device-dependent error 6) Query error 7) Request for bus control right 8) Operation complete The Logical OR output bit is represented by Status Byte Register bit 5 (DIO6) as a summary message for the Event Status Bit (ESB).	The Status Byte Register is a register in which the RQS bit and the seven summary message bits from the status data structure can be set. This register is used together with the Service Request Enable Register. When the results of the OR operation of both register contents are other than 0, SRQ becomes ON. To indicate this, bit 6 of the Status Byte Register (DIO7) is reserved by the system as the RQS bit. The RQS bit is used to indicate that there is a service request for the external controller. The mechanism of SRQ conforms to the IEEE488.1 standard.	The Output Queue has the structure of the queue model described above. Status Byte Register bit 4 (DIO5) is set as a summary message for Message Available (MAV) to indicate that there is data in the output buffer.

Status Byte (STB) Register

The STB register consists of the STB and RQS (or MSS) messages of the device.

ESB and MAV summary messages

This paragraph describes the ESB and MAV summary messages.

(1) ESB summary message

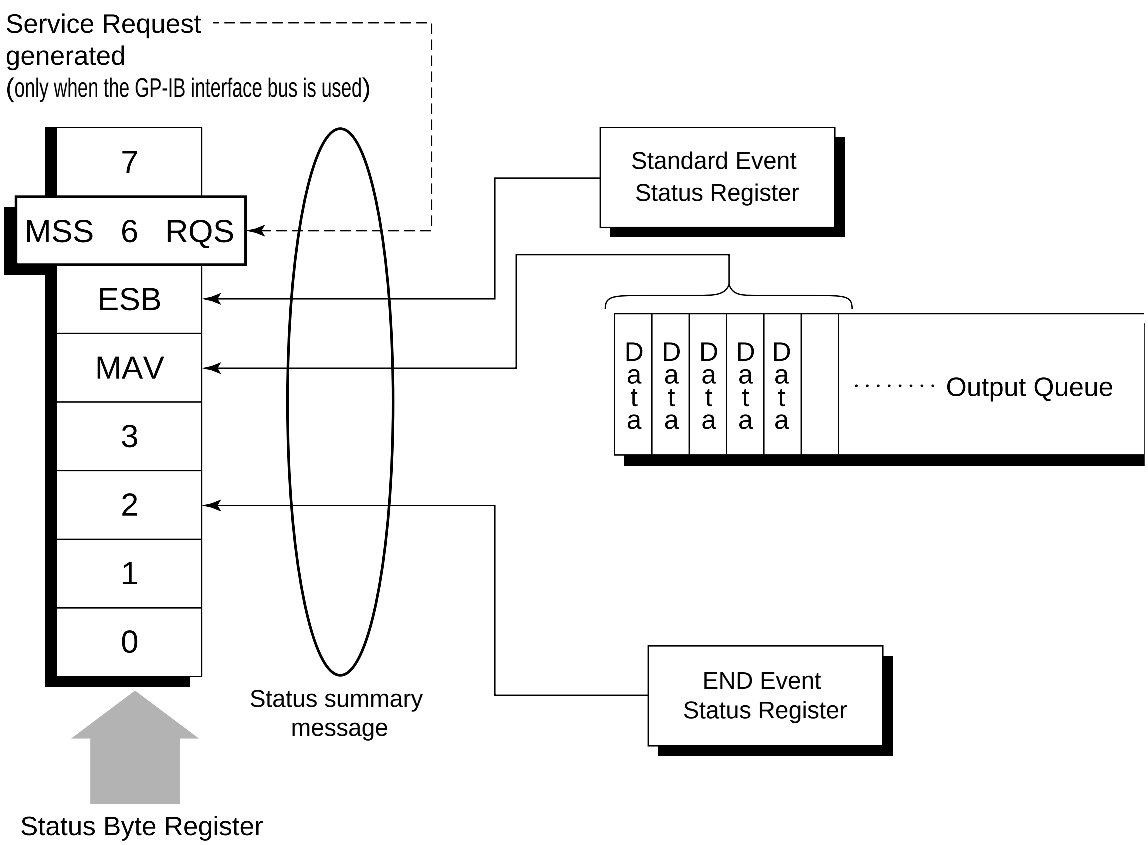
The ESB (Event Summary Bit) is a message defined by IEEE488.2 which uses bit 5 of the STB register. When the setting permits events to occur, the ESB summary message bit becomes 1 if any one of the events recorded in the Standard Status Register becomes 1. Conversely, the ESB summary message bit becomes 0 if one of the recorded events occurs, even if events are set to occur.
This bit becomes 0 when the ESR register is read by the *ESR? query or when it is cleared by the *CLS command.

(2) MAV summary message

The MAV (Message Available) summary bit is a message defined by IEEE488.2 which uses bit 4 of the STB register. This bit indicates whether the output queue is empty. The MAV summary message bit is
set to 1 when a device is ready to receive a request for a response message from the controller. When the output queue is empty, this bit is set to 0. This message is used to synchronize the information exchange with the controller. For example, this message is available when, after the controller sends a query command to a device, the controller waits until MAV becomes 1. While the controller is waiting for a response from the device, other jobs can be processed. Reading the Output Queue without first checking MAV will cause all system bus operations to be delayed until the device responds.

Device-dependent summary messages

As shown in the diagram below, the spectrum analyzer does not use bits 0, 1, 3, and 7, and it uses bit 2 as the summary bit of the Event Status Register.



Reading and clearing the STB register

The STB register can be read using serial polling or the ∗STB? common query. The IEEE488.1 STB message can be read by either method, but the value sent to bit 6 (position) is different for each method.
The STB register contents can be cleared using the ∗CLS command.

(1) Reading by serial polling (only when the GP-IB interface bus is used)

The IEEE488.1 serial polling allows the device to return a 7-bit status byte and an RQS message bit which conforms to IEEE488.1. The value of the status byte is not changed by serial polling. The device sets the RQS message to 0 immediately after being polled.

(2) Reading by the ∗STB? common query

The ∗STB? common query requires the devices to send the contents of the STB register and the integer format response messages, including the MSS (Master Summary Status) summary message. Therefore, except for bit 6, which represents the MSS summary message, the response to ∗STB? is identical to that of serial polling.

(3) Definition of MSS (Master Summary Message)

MSS indicates that there is at least one cause for a service request. The MSS message is represented at bit 6 response to an ∗STB? query, but it is not produced as a response to serial polling. It should not be taken as part of the status byte specified by IEEE488.1. MSS is configured by the overall logical OR in which the STB register and SRQ enable (SRE) register are combined.

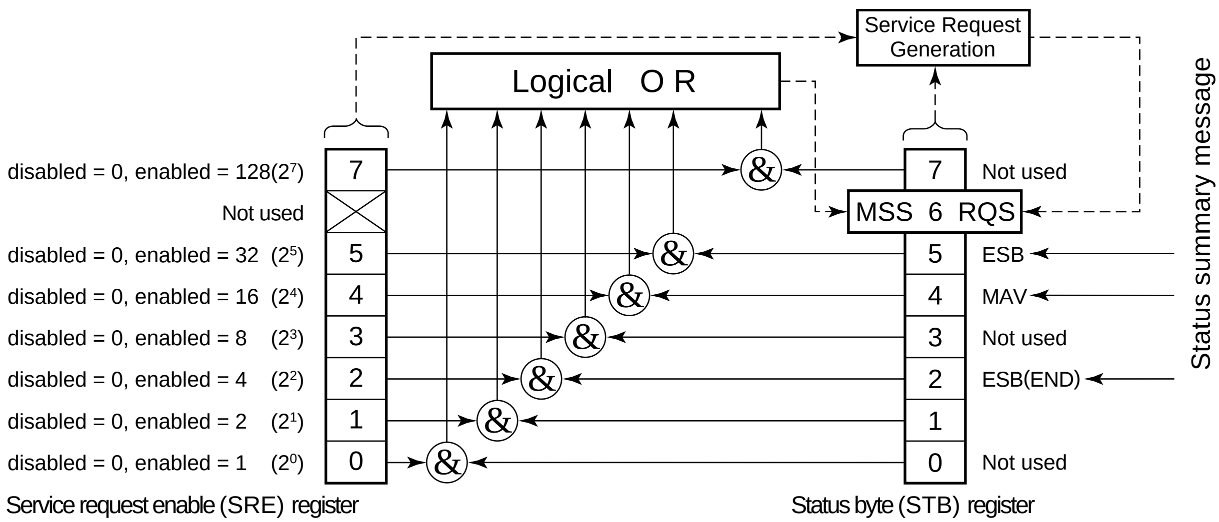
(4) Clearing the STB register using the ∗CLS common command

The ∗CLS common command clears all status data structures as well as the summary messages corresponding to them.
The ∗CLS command does not affect the settings in the Enable Register.

Service Request (SRQ) Enabling Operation

Bits 0 to 7 of the Service Request Enable Register (SRE) determine which bit of the corresponding STB register can generate SRQ.

The bits in the Service Request Enable Register correspond to the bits in the Status Byte Register. If a bit in the Status Byte Register corresponding to an enabled bit in the Service Request Enable Register is set to 1, the device makes a service request to the controller with the RQS bit set to 1.



(1) Reading the SRE register

The contents of the SRE register are read using the *SRE? common query. The response message to this query is an integer from 0 to 255 which is the sum of the bit digit weighted values in the SRE register.

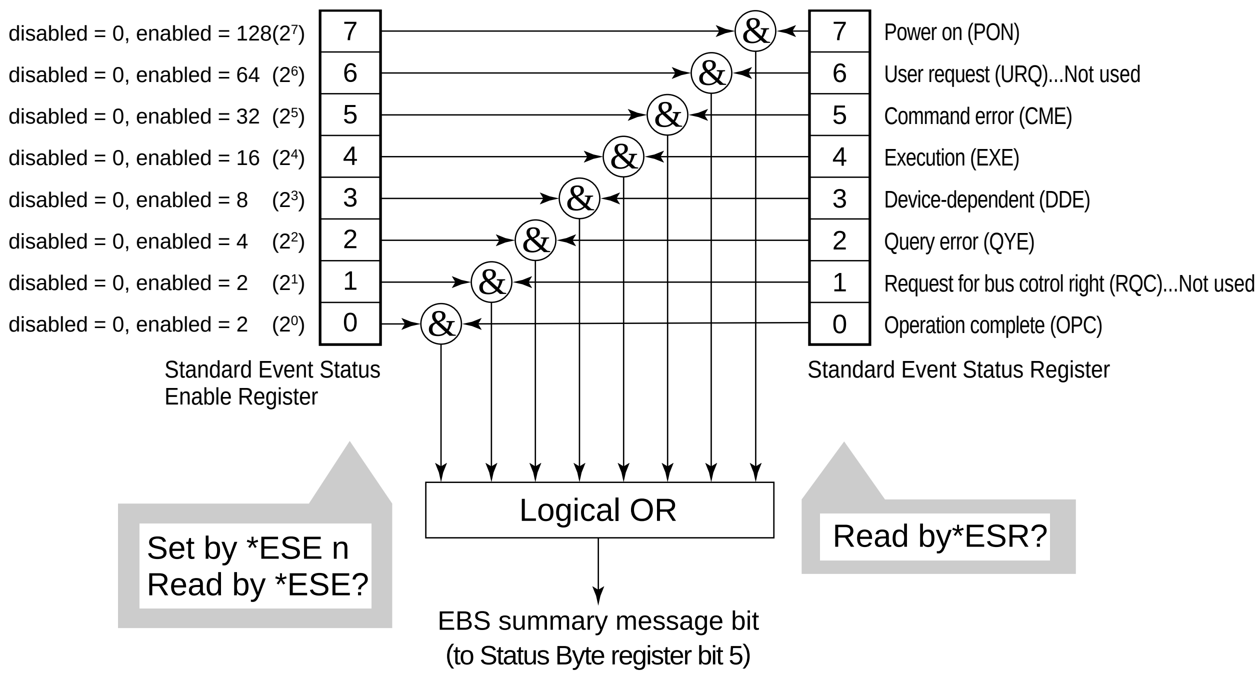
(2) Updating the SRE register

The SRE register is written using the *SRE common command. An integer from 0 to 255 is assigned as a parameter to set the SRE register bit to 0 or 1. The value of bit 6 is ignored.

Standard Event Status Register

Bit definition of Standard Event Status Register

The diagram below shows the operation of the Standard Event Status Register.



The Standard Event Status Enable (ESE) Register on the left is used to select which bits in the corresponding Event Register will cause a TRUE summary message when set.

Bit	Event name	Description
7	Power on (PON-Power on)	A transition from power-off to power-on occurred during the power-up procedure.
6	Not used	
5	Command error (CME-Command Error)	An illegal program message or a misspelled command was received.
4	Execution error (EXE-Execution Error)	A legal but unexecutable program message was received.
3	Device-dependent error (DDE-Device-dependent Error)	An error not caused by CME, EXE, or QYE occurred (parameter error, etc.).
2	Query error (QYE-Query Error)	An attempt was made to read data in the Output Queue when it was empty. Or, the data in the Output Queue was lost before it was read.
1	Not used	
0	Operation complete (OPC-Operation Complete)	This bit becomes 1 when this instrument has processed the *OPC command.

Reading, writing, and clearing the Standard Event Status Register

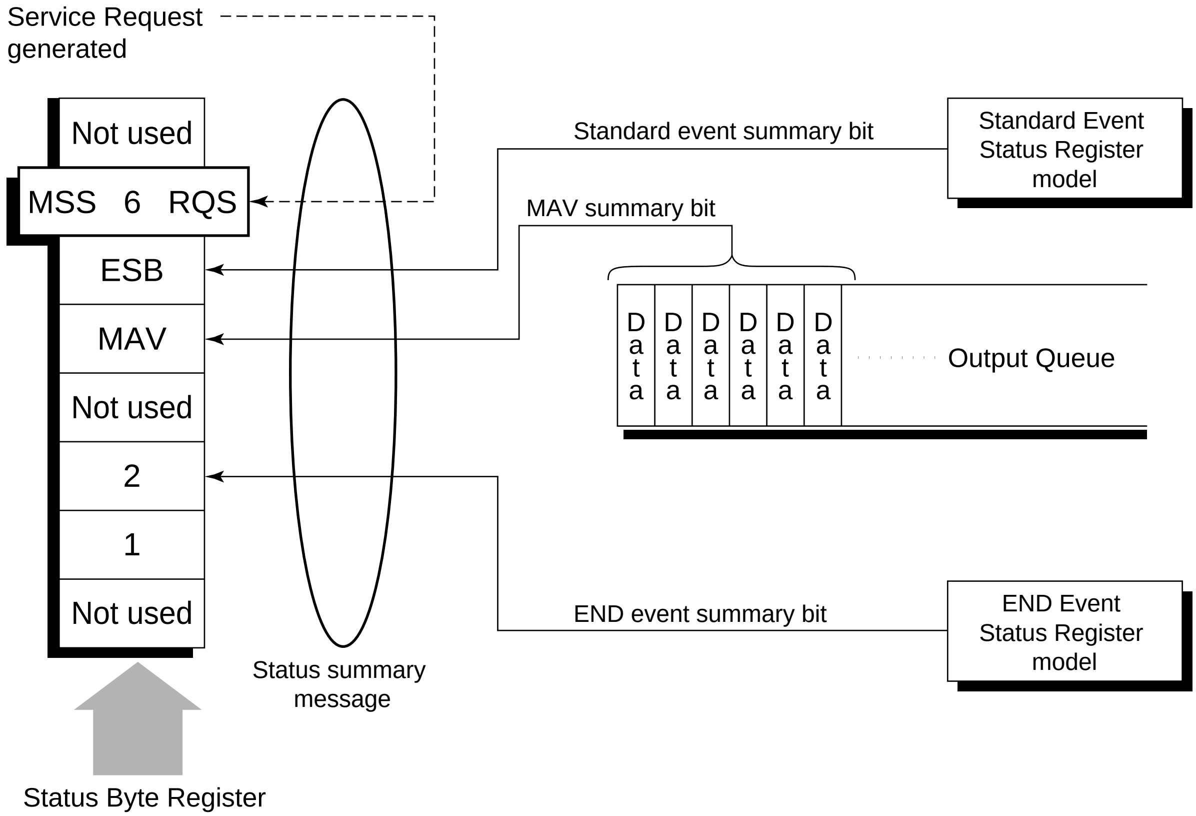
Reading	The register is read using the ∗ESR? command query. The register is cleared after being read. The response message is integer-format data with the binary weight added to the event bit and the sum converted to decimal.
Writing	With the exception of clearing, data cannot be written to the register from outside.
Clearing	The register is cleared when: ① A ∗CLS command is received ② The power is turned on Bit 7 is set to ON, and the other bits are cleared to 0 ③ An event is read for the ∗ESR? query command

Reading, writing, and clearing the Standard Event Status Enable Register

Reading	The registers is read using the ∗ESE? command. The response message is integer-format data with the binary weight added to the event bit and the sum converted to decimal.
Writing	The register is written using the ∗ESE common command.
Clearing	The register is cleared when: ① An ∗EXE command with a data value of 0 is received ② The power is turned on The Standard Event Enable Register is not affected when: ① The device clear function status of IEEE488.1 is changed ② An ∗RST common command is received ③ A ∗CLS common command is received

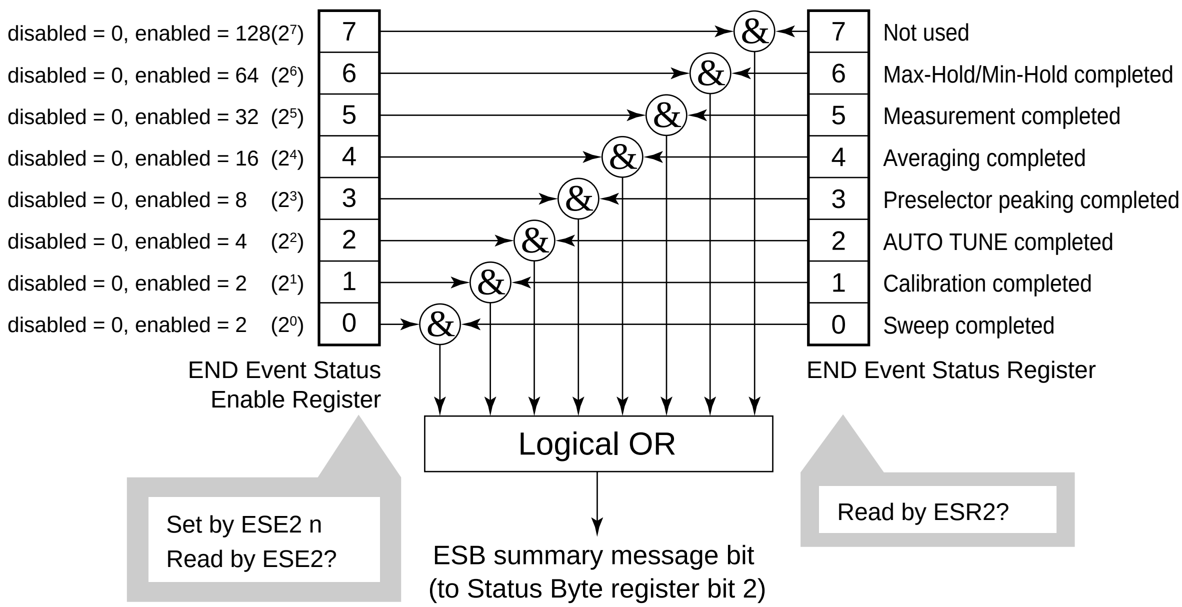
Extended Event Status Register

For the MS2665C/67C/68C, bits 7, 3, 1, and 0 are unused. Bit 2 is assigned to the END summary bit as the status-summary bit supplied by the extended register model as shown below.



Bit definition of END Event Status Register

The diagram below shows the operation and event-bit names of the END Event Status Register.



The END Event Status Enable Register on the left is used to select which bits in the corresponding Event Register will cause a TRUE summary message when set.

Bit	Event name	Description
7	Not used	Not used
6	Max Hold/Min Hold	Sweeping according to the specified HOLD number has been completed.
5	Measurement completed	Calculation processing for measurements (frequency count, noise, etc.) has been completed.
4	Averaging completed	Sweeping according to the specified AVERAGE number has been completed.
3	Preselector peaking completed	Preselector peaking has been completed
2	AUTO TUNE completed	AUTO TUNE has been completed.
1	Calibration completed	ALL CAL, LEVEL CAL, or FREQ CAL has been completed.
0	Sweep completed	A single sweep has been completed or is in standby.

Reading, writing, and clearing the Extended Event Status Register

Reading	The ESR? common query is used to read the register. The register is cleared after being read. The response message is integer-format data with the binary weight added to the event bit and the sum converted to decimal.
Writing	With the exception of clearing, data cannot be written to the register from outside.
Clearing	The register is cleared when: ① A *CLS command is received ② The power is turned on ③ An event is read for the ESR2? query command

Reading, writing, and clearing the Extended Status Enable Register

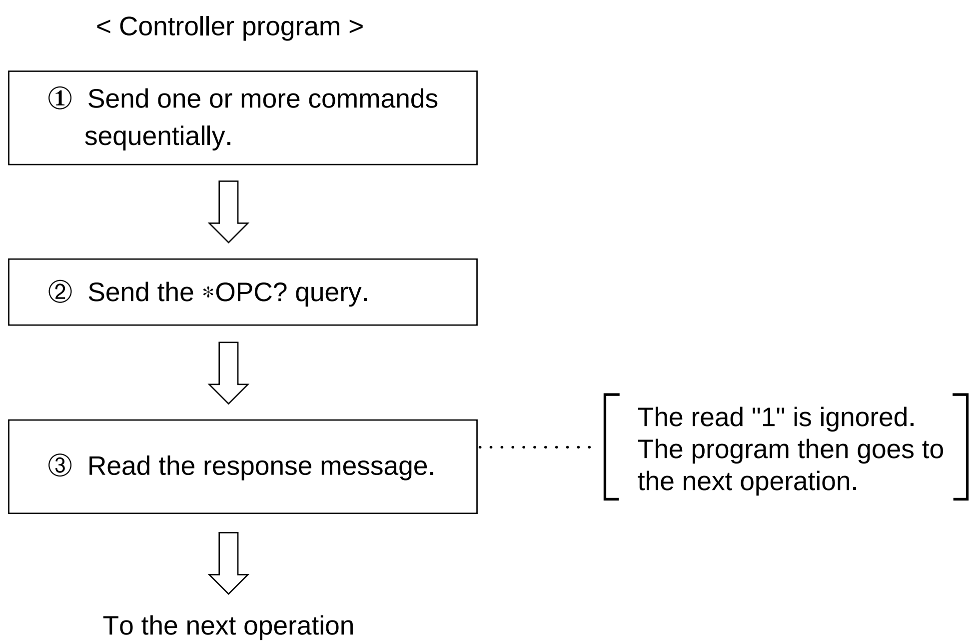
Reading	The ESE2? query is used to read the register. The response message is integer-format data with the binary weight added to the event bit and the sum converted to decimals.
Writing	The ESE2 program command is used to write the register. Because bits 0 to 7 of the registers are weighted with values 1, 2, 4, 8, 16, 32, 64, and 128, respectively, the write data is transmitted as integer-format data that is the sum of the requiredbit digits selected from the weighted value.
Clearing	The register is cleared when: ① An ESE2 program command with a data value of 0 is received ② The power is turned on The Extended Event Status Enable register is not affected when: ① The device clear function status of IEEE488.1 is changed ② An *RST common command is received ③ A *CLS common command is received

Techniques for Synchronizing MS2665C/67C/68C with a Controller

The MS2665C/67C/68C usually treats program messages as sequential commands that do not process newly-received commands until they complete the processing of the previous command. Therefore, no special consideration is necessary for pair-synchronization between the MS2665C/67C/68C and the controller.
If the controller controls and synchronizes with one or more devices, after all the commands specified for the MS2665C/67C/68C have been processed, the next commands must be sent to other devices.
There are two ways of synchronizing the MS2665C/67C/68C with the controller:
① Wait for a response after the ∗OPC? query is sent.
② Wait for SRQ after ∗OPC is sent.

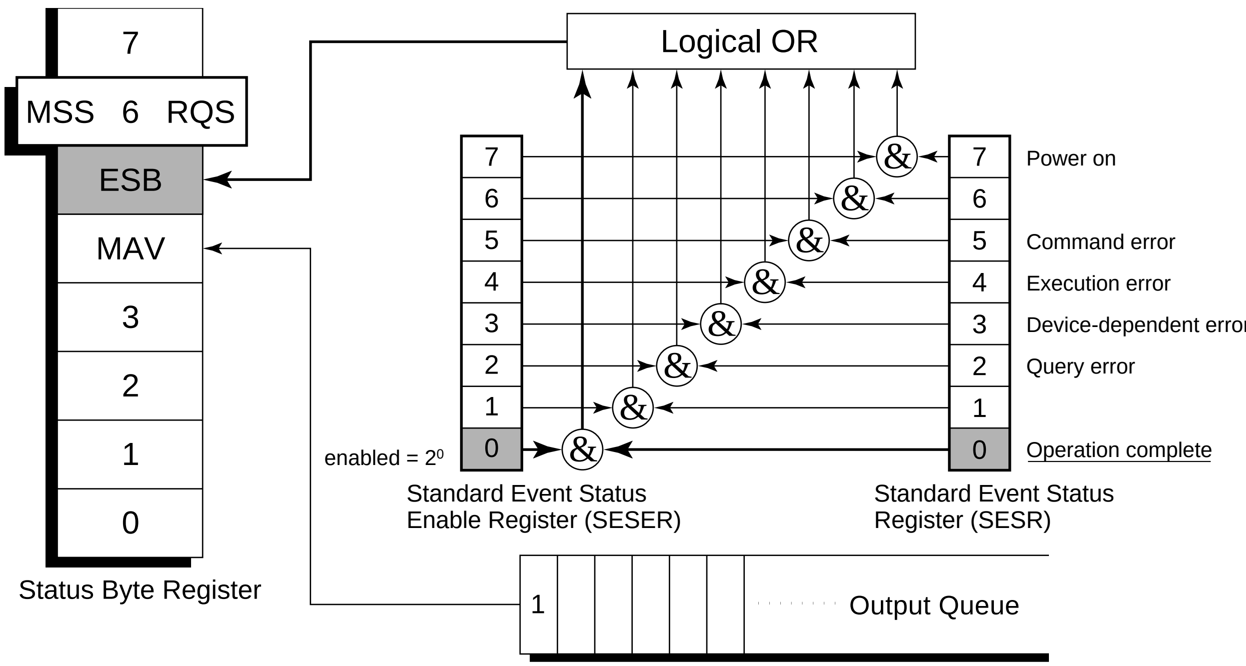
Wait for a response after the ∗OPC? query is sent.

The MS2665C/67C/68C outputs "1" as the response message when executing the *OPC? query command. The controller is synchronized with the MS2665C/67C/68C by waiting for the response message to be entered.



Wait for a service request after ∗OPC is sent (only when the GP-IB interface bus is used).

The MS2665C/67C/68C sets the operation-complete bit (bit 0) to 1 when executing the *OPC command. The controller is synchronized with the spectrum analyzer for SRQ when the operation-complete bit is set for SRQ.



■ < Controller program >

① Enable the 2^0 bit of the Standard Event Status Enable Register.	`PRINT @1;"∗ESE 1"`
⇩	
② Enable the 2^5 bit of the Service Request Enable Register.	`PRINT @1;"∗SRE 32"`
⇩	
③ Make the device execute the specified operation.	
⇩	
④ Send the ∗OPC command.	`PRINT @1;"∗OPC"`
⇩	
⑤ Wait for the SRQ interrupt (ESB summary message).	…… Value of status byte: $2^6 + 2^5 = 96$

SECTION 5

INITIAL SETTINGS

The MS2665C/67C/68C initializes the GP-IB interface system at three levels in accordance with the IEEE488.2 specifications. This section describes how these three levels of initialization are processed, and how to instruct initialization from the controller.

TABLE OF CONTENTS

SECTION 5
INITIAL SETTINGS

In the IEEE488.2 standard, there are three levels of initialization. The first level is "bus initialization," the second level is "initialization for message exchange," and the third level is "device initialization." This standard also stipulates that a device must be set to a known state when the power is turned on.

Level	Initialization type	Description	Level combination and sequence
1	Bus initialization	The IFC message from the controller initializes all interface functions connected to the bus.	Level 1 can be combined with other levels, but must be executed before level 2.
2	Initialization for message exchange	Message exchanges of all devices and specified devices on the GP-IB are initialized using the SDC and DCL GP-IB bus commands, respectively. These commands also nullify the function that reports operation completion to the controller.	Level 2 can be combined with other levels, but must be executed before level 3.
3	Device initialization	The ∗RST or INI/IP command returns a specified device to a known device-specific state, regardless of the conditions under which it was being used.	Level 3 can be combined with other levels, but must be executed after levels 1 and 2.

When using the standard RS-232C interface port to control the MS2665C/67C/68C from the controller, the level-3 device initialization function of can be used, and the level-2 initialization function cannot be used. When using the GP-IB interface bus to control the MS2665C/67C/68C from the controller, the initialization functions of levels 1, 2, and 3 can be used.

The following paragraph describes the commands for initialization at levels 1, 2, and 3 and the items that are initialized. This paragraph also describes the known state which is set when the power is turned on.

Bus Initialization using the IFC Statement

■ Example

```
board% = 0
CALL SendIFC (board%)
```

■ Explanation

This function can be using when using the GP-IB interface bus is used to control the spectrum analyzer from the controller.
The IFC statement initializes the interface functions of all devices connected to the GP-IB bus line.
The initialization of interface functions involves clearing the interface function states of devices set by the controller, and resetting them to their initial states. In the table below, indicates the functions which are initialized, and indicates the functions which are partially initialized.

No	Function	Symbol	Initialization by IFC
1	Source handshake	SH	○
2	Acceptor handshake	AH	○
3	Talker or extended talker	T or TE	○
4	Listener or extended listener	L or LT	○
5	Service request	SR	△
6	Remote/local	RL	
7	Parallel poll	PP	
8	Device clear	DC	
9	Device trigger	DT	
10	Controller	C	○

Bus initialization by the IFC statement does not affect the device operating state (frequency settings, LED on/off, etc.).

Initialization for Message Exchange by DCL and SDC Bus Commands

Example

Initializes all devices on the bus for message exchange (sending DCL).

```
board% = 0
addresslist% = NOADDR
CALL DevClearList(board%, addresslist%)
```

Initializes only the device at address 3 for message exchange (sending SDC).

```
board% = 0
address% = 3
CALL DevClear(board%, address%)
```

Explanation

This function can be used when the GP-IB interface is used to control the spectrum analyzer from the controller. This statement executes initialization for message exchange of all devices or a specified device on the GP-IB having the specified select code.

Items to be initialized for message exchange

When the spectrum analyzer accepts the DCL or SDC bus command, it does the following:

(1) Input buffer and Output Queue:	Clears them and also clears the MAV bit.
(2) Parser, Execution Controller, and Response Formatter:	Resets them.
(3) Device commands including ∗RST:	Clears all commands that prevent these commands from being executed.
(4) Processing of the ∗OPC? command:	Puts a device in OCIS (Operation Complete Command Idle State). As a result, the operation complete bit cannot be set in the Standard Event Status Register.
(5) Processing of the ∗OPC? query:	Puts a device in OQIS (Operation Complete Query Idle State). As a result, the operation complete bit 1 cannot be set in the Output Queue.
(6) Device functions:	Puts all functions associated with message exchange in the idle state. The device continues to wait for a message from the controller.

CAUTION

The following are not affected even if the DCL and SDC commands are processed.

(1) Current data set or stored in the device
(2) Front panel settings
(3) Status of status byte other than MAV bit
(4) device operation in progress

Device Initialization using the ∗RST Command

■ Syntax

```
∗RST
```

■ Example

For RS-232C

`WRITE #1,"*RST"` .... Initializes the device (Spectrum Analyzer) at address 1 at level 3.

For GPIB

```
SPA%=1
CALL Send(0,SPA,"*RST",NLend)
```

■ Explanation

The ∗RST (Reset) command is an IEEE488.2 common command that resets a device at level 3.
The ∗RST (Reset) command is used to reset a device (Spectrum Analyzer) to a specific initial state. For details of the items that are initialized and the settings after initialization, see Appendix A.

Note: The ∗RST command does not affect the following.

(1) IEEE488.1 interface state
(2) Device address
(3) Output Queue
(4) Service Request Enable register
(5) Standard Event Status Enable register
(6) Power-on-status-clear flag setting
(7) Calibration data affecting device specifications
(8) Parameters preset for control of external device, etc.

☞ For details of the settings of the spectrum analyzer after initialization, see Appendix A.

Device Initialization using the INI/IP Command

■ Syntax

```
INI
IP
```

■ Example (program message)

For RS-232C

`WRITE #1,"INI"` .... Initializes the device (Spectrum Analyzer) at address 1 at level 3.

For GPIB

```
SPA%=1
CALL Send(0,SPA%,"INI",NLend)
```

■ Explanation

The INI and IP commands are the MS2665C/67C/68C device-dependent messages that initialize a device at level 3.

For details of the items that are initialized by the INI and IP commands, and the settings after initialization, see Appendix A.

Device Status at Power-on

When the power is turned on:

(1) The device is set to the status it was in at power-off.
(2) The Input Buffer and Output Queue are cleared.
(3) The Parser, Execution Controller, and Response Formatter are initialized.
(4) The device is put into OCIS (Operation Complete Command Idle State).
(5) The device is put into OQIS (Operation Complete Query Idle State).
(6) The Standard Event Status and Standard Event Status Enable Registers are cleared. Events can be recorded after the registers have been cleared.

As the special case of (1), when the spectrum analyzer is powered on for the first time after delivery, the spectrum analyzer settings are those listed in the Initial Settings Table(☞ Appendix A).

SECTION 6
SAMPLE PROGRAMS

This section gives some examples of the Microsoft Quick Basic program that controls the MS2665C/67C/68C from a personal computer which is used as a controller.
Note: Microsoft Quick Basic is a trade mark of the Microsoft Corporation.

TABLE OF CONTENTS

SECTION 6 SAMPLE PROGRAMS

Precautions on Creating the Remote Control Program

Note the following points when writing remote control programs.

No.	Precaution	Description
1	Be sure to initialize each device.	each device. When a command other than the INPUT #statement is sent to the controller before the response to a query is read, the output buffer is cleared, and the response message disappears. For this reason, write the INPUT #statement in immediate succession to a query.
2	Do not send any command (related to the device) other than the INPUT #statement immediately after sending a query.	No.2 described above is one type of exception processing of the protocol. Avoid exception processing from occurring as requested. Avoid stoppage of execution caused by an error by providing a program with exception-processing section against exceptions that can be foreseen.
3	Create a program that avoids the exception processing of the protocol.	There may be a number of the state in which each device is not proper to be actually sued due to operation on its own panel or execution of other programs. It is necessary to using individual devices with a prescribed condition resulting from initializing them. Execute initialization (INIT or *RST) of the functions proper to
4	Protect RS-232C buffer overflow.	The RS-232C interface has a 512-byte data area as the internal receive buffer. The buffer overflow may occur depending on the processing. To protect the overflow, don't send a large amount of data(i.e. control commands) at a time for remote control using RS-232C. After sendind a command group, send *OPC? command to check the response for the synchronization before sending the next command.

Sample Programs

Initializing

<Example 1> Initializes the MS2660 series

```
'+++++++++++++++++++++++++++++++++++++
' MS2660 series Sample program
'  <<Initialize>>
'+++++++++++++++++++++++++++++++++++++
'
' Setup parameter of PC Com. port
'   BAUD       :24ØØ BPS
'   Parity     : NONE
'   Data bit   : 8 bits
'   Stop bit   : 1 bit
'   Terninator : LineFeed
'
OPEN "COM1:24ØØ,N,8,1,CD5ØØ,DSØ,LF" FOR RANDOM AS #1
'
PRINT #1, "INI"'   Initialize MS2660 series Spectrum Analizer
'
END
```

The parameters initialized by the above program are shown in Appendix A.

There is a '∗RST' command in another command for executing initialization. The '∗RST' command is used to execute initialization over a wider range. For the range of initialization level, see SECTION 5. The usage of the 'IP' command is identical to the 'INI' command.

For general usage of INI and ∗RST, first initialize the MS2665C/67C/68C device functions with the IP or INI command, then use the program commands to set only the functions to be changed. This prevents the spectrum analyzer from being controlled while unnecessary functions are set.

Reading the frequency and level at marker point

<Example 2> Sets the center frequency to 500 MHz and span to 10 MHz, then displays the frequency and level reading at the peak point on the controller screen when a signal to be measured is received.

```
 1 '+++++++++++++++++++++++++++++++++++++++++++
 2 ' MS2660 series Sample program
 3 '  <<Read out marker frequency & level>>
 4 '+++++++++++++++++++++++++++++++++++++++++++
 5 '
 6 ' Setup parameter of PC Com. port
 7 '
 8 OPEN "COM1:24ØØ,N,8,1,CD5ØØ,DSØ,LF" FOR RANDOM AS #1
 9 '
1Ø PRINT #1, "INI"'          Initialize Spectrum Analyzer
11 '
12 PRINT #1, "CF 5ØØMHZ"'   Center fequency :5ØØMHz
13 PRINT #1, "SP 1ØMHZ"'    Span frequency  :1ØMHz
14 PRINT #1, "TS"'          Take a sweep
15 '
16 PRINT #1, "PCF"'         Set peak to center frequency
17 PRINT #1, "PRL"'         Set peak to reference level
18 PRINT #1, "MKPK"'        Search peak
19 '
2Ø PRINT #1, "MKF?"'        Query marker frequency
21 INPUT #1, FREQ'          Input marker frequency data
22 PRINT #1, "MKL?"'        Query marker level
23 INPUT #1, LEVEL'         Input marker level data
24 '
25 '                        Print out the result(Frequency/Level)
26 PRINT USING "Marker  Frequency=####.### MHz";FREQ/1ØØØØØØ
27 PRINT USING "Marker  LEVEL=####.##  dBm";LEVEL
28 '
29 END
```

The center frequency and frequency span are set at line 12 and line 13 respectively. The TS sweep command at line 14 does not execute the next message unless the sweep is completed. This command thus prevents the peak search and other program lines from being executed before the sweep is completed.

The PCF and PRL commands at lines 16 and 17 operate as follows: The former sets the peak point on the screen to the center frequency, and the latter sets its peak level center frequency to the reference level.

The "MKF?" and "MKL?" at lines 20 and 22 query the frequency and level at the marker point respectively, and the data is read with the INPUT#statement on the next line. When a command other than the INPUT#statement is sent before the response to a query is read, the output buffer is cleared, and the response message is deleted. For this reason, write the INPUT#statement immediately after a query.

Program execution result of <Example 2>

Marker Frequency=501.251△MHz

Marker LEVEL=–15.53dBm

Note: △is a space.

Reading trace data

<Example 3-1> Reads the trace level at all points when CF and SPAN are set to 500 MHz and 10 MHz respectively.

```
 1 '+++++++++++++++++++++++++++++++++++++
 2 ' MS2660 series Sample program
 3 '  <<Read out trace data(ASCII)>>
 4 '+++++++++++++++++++++++++++++++++++++
 5 '
 6 ' Setup parameter of PC Com. port
 7 '
 8 OPEN "COM1:24ØØ,N,8,1,CD5ØØ,DSØ,LF" FOR RANDOM AS #1
 9 '
1Ø PRINT #1, "INI"'     Initialize Spectrum Analyzer
11 '
12 PRINT #1, "CF 5ØØMHZ"'    Center fequency :5ØØMHz
13 PRINT #1, "SP 1ØMHZ"'     Span frequency  :1ØMHz
14 PRINT #1, "TS"'           Take a sweep
15 '
16 DIM TRACE(5Ø1)'           Define read data area
17 PRINT #1, "BIN Ø"'        Set read out data type to ASCII
18 '
19 FOR I = Ø TO 5ØØ'         Repeat trace(Ø) to trace(5ØØ):5Ø1 points
2Ø PRINT #1, "XMA? " + STR$(I) + ",1"'   Query trace data
21 INPUT #1, TRACE(I)'       Read out trace data
22 '                         Print out trace data
23 PRINT USING "###.##dBm"; TRACE(I) / 1ØØ
24 NEXT I
25 '
26 END
```

The "BIN_0" at line 17 is a command for specifying ASCII as the response data format. The ASCII or BINARY transfer format can be specified for the "XMA?", "XMB?", "XMG?", and "XMT?" queries for reading trace data.

The example 3-2 blocks the trace data at every 10 points, and reads it.

<Example 3-2> Blocks the trace data at every 10 points, and reads it.

```
 1 '+++++++++++++++++++++++++++++++++++++
 2 ' MS2660 series Sample program
 3 '  <<Read out trace data(ASCII) BLOCKING>>
 4 '+++++++++++++++++++++++++++++++++++++
 5 '
 6 ' Setup parameter of PC Com. port
 7 '
 8 OPEN "COM1:24ØØ,N,8,1,CD5ØØ,DSØ,LF" FOR RANDOM AS #1
 9 '
1Ø PRINT #1, "INI"'     Initialize Spectrum Analyzer
11 '
12 PRINT #1, "CF 5ØØMHZ"'    Center fequency :5ØØMHz
13 PRINT #1, "SP 1ØMHZ"'     Span frequency  :1ØMHz
14 PRINT #1, "TS"'           Take a sweep
15 '
16 DIM TRACE(5Ø1)'           Define read data area
17 PRINT #1, "BIN Ø"'        Set read out data type to ASCII
18 '
19 FOR I = Ø TO 49Ø STEP 1Ø
2Ø                           Repeat trace(Ø) to trace(499):5ØØ points
21                           Blocking 1Ø trace data
22     PRINT #1, "XMA? " + STR$(I) + ",1Ø"'   Query trace data
23                                            Read out trace data
24      INPUT #1, TRACE(I), TRACE(I + 1), TRACE(I + 2), TRACE(I + 3),
TRACE(I + 4), TRACE(I + 5), TRACE(I + 6), TRACE(I + 7), TRACE(I + 8),
TRACE(I + 9)
25     PRINT TRACE(I), TRACE(I + 1), TRACE(I + 2), TRACE(I + 3), TRACE(I
+ 4), TRACE(I + 5), TRACE(I + 6), TRACE(I + 7), TRACE(I + 8),TRACE(I + 9)
26 NEXT I
27 PRINT #1, "XMA? 5ØØ,1"'  Query last trace data:trace(5ØØ)"
28 INPUT #1, TRACE(5ØØ)
29'
3Ø FOR I = Ø TO 5ØØ'         Print out trace data
31     PRINT USING "###.##dBm"; TRACE(I) / 1ØØ
32 NEXT I
33 '
34 END
```

Delta marker

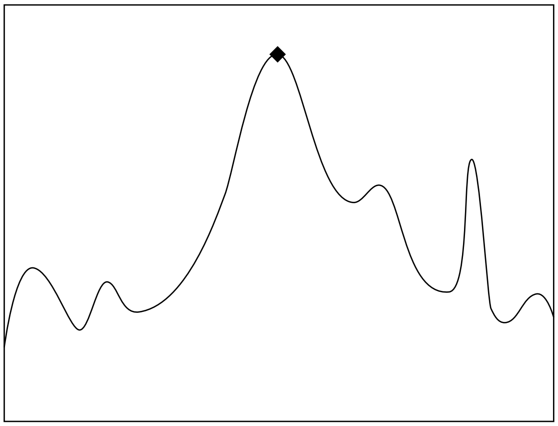
<Example 4> Using a delta marker, reads out the frequency and level differences between a peak point and the next peak point.

```
 1 '+++++++++++++++++++++++++++++++++++++++++++++++
 2 ' MS2660 series Sample program
 3 '  <<Read out delta marker frequency & level>>
 4 '+++++++++++++++++++++++++++++++++++++++++++++++
 5 '
 6 ' Setup parameter of PC Com. port
 7 '
 8 OPEN "COM1:24ØØ,N,8,1,CD5ØØ,DSØ,LF" FOR RANDOM AS #1
 9 '
1Ø PRINT #1, "INI"'          Initialize Spectrum Analyzer
11 '
12 PRINT #1, "FA 5ØMHZ"'     Start fequency :5ØØMHz"
13 PRINT #1, "FB 2GHZ"'      Stop frequency   :2GHz
14 PRINT #1, "TS"'           Take a sweep
15 '
16 PRINT #1, "MKR Ø"'        Set marker to "Normal"
17 PRINT #1, "MKPK"'         search peak
18 PRINT #1, "MKR 1"'        Set marker to "Delta"
19 PRINT #1, "MKPK NH"'      search Next peak
2Ø '
21 PRINT #1, "MKF?"'         Query Delta marker frequency
22 INPUT #1, DFREQ'          Input Delta marker frequency data
23 PRINT #1, "MKL?"'         Query Delta marker level
24 INPUT #1, DLEVEL'         Input Delta marker level data
25 '                         Print out the result(Frequency/Level)
26 PRINT USING "Delta Frequency=####.### MHz"; DFREQ / 1ØØØØØØ
27 PRINT USING "Delta     level=####.##  dB"; DLEVEL
28 '
29 END
```

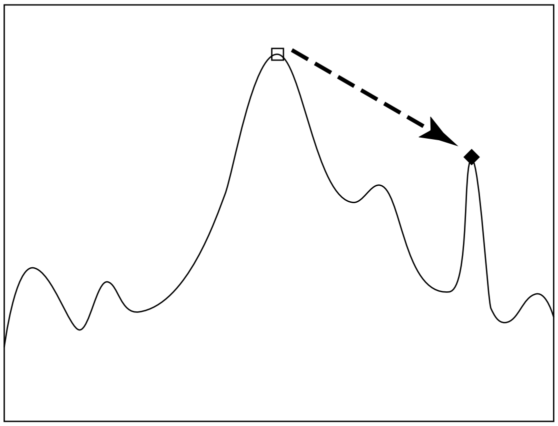
The "MKR_1" at line 18 is used to set the marker mode to DELTA, so that the reference marker can also be set together to the current marker position.

The "MKPK_NH" at line 19 sets the marker search to NEXT PEAK to move the current marker to NEXT PEAK point.

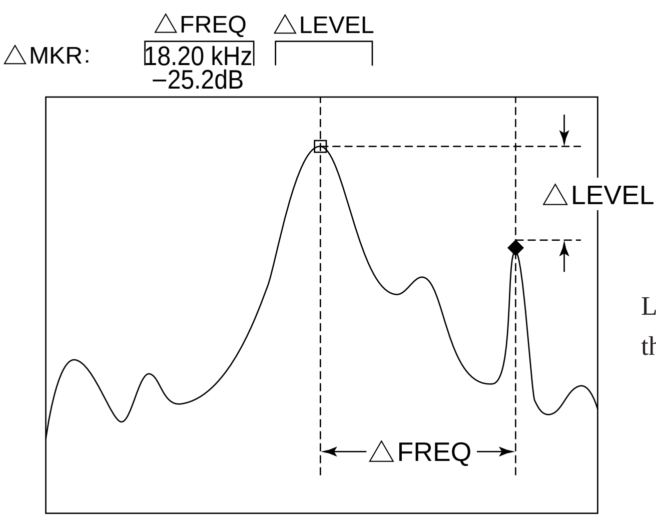
The "MKF?" and "MKL?" at lines 21 and 23 query reading the frequency and level at the current marker position while the marker mode is NORMAL. It is also used to query reading the frequency and level differences between the current marker and the reference marker while the marker mode is DELTA.



Executing PEAK SEARCH (MKPK) at line 17 allows the current marker to be set to the peak point.



Line 19 allows the reference marker to be set together to the current marker position. Executing NEXT PEAK SEARCH MKPK_NH at line 18 allows the current marker



Lines 21 to 24 read out the FREQ and LEVEL displayed in the upper left of screen.

Multimarker function

<Example 5-1> Using the multimarker function, measures the frequency/level at 10 points in descending order.

```
 1 '++++++++++++++++++++++++++++++++++
 2 ' MS2660 series Sample program
 3 '  <<Multi Marker Highest-1Ø>>
 4 '++++++++++++++++++++++++++++++++++
 5 '
 6 ' Setup parameter of PC Com. port
 7 '
 8 OPEN "COM1:24ØØ,N,8,1,CD5ØØ,DSØ,LF" FOR RANDOM AS #1
 9 '
10 PRINT #1, "INI"'          Initialize Spectrum Analyzer
11 '
12 PRINT #1, "CF 5ØØMHZ"'    Center fequency 5ØØMHz
13 PRINT #1, "SP 2ØKHZ"'     Span frequency  2ØKHz
14 PRINT #1, "TS"'           Take a sweep
15 '
16 PRINT #1, "MKMHI"'        Multi marker On &
17 '                         Perform Highest-1Ø function
18 '
19 FOR I = 1 TO 1Ø
20 PRINT #1, "MKMP? " + STR$(I)
21 INPUT #1, FREQ'           Input marker frequency data
22 PRINT #1, "MKML? " + STR$(I)
23 INPUT #1, LEVEL'          Input marker frequency data
24 '
25  PRINT USING "Marker No. ## #,###.####MHz ####.##dBm"; I; FREQ / 1ØØØØØØ;
LEVEL
26 NEXT I
27 '
28 END
```

The MS2665C/67C/68C multimarker function allows up to ten markers to be set at a time. The "MKMHI" at line 130 is used to set the multimarker to HIGHEST 10 mode which sets up to ten markers in descending order.

The frequency and level at each marker are read out by lines 19 to 26.

This program allows harmonics to be observed if the program is modified. <Example 5-2> shows the program for observing the harmonics from a fundamental to the fifth order.

<Example 5-2> Harmonic frequency measurement (measures 500 MHz fundamental and up to its fifth order harmonics)

```
 1 '+++++++++++++++++++++++++++++++++
 2 ' MS2660 series Sample program
 3 '  <<Multi Marker Harmonics>>
 4 '+++++++++++++++++++++++++++++++++
 5 '
 6 ' Setup parameter of PC Com. port
 7 '
 8 OPEN "COM1:24ØØ,N,8,1,CD5ØØ,DSØ,LF" FOR RANDOM AS #1
 9 '
1Ø PRINT #1, "INI"'          Initialize Spectrum Analyzer
11 '
12 PRINT #1, "FA ØHZ"'       Start fequency :ØHz
13 PRINT #1, "FB 3GHZ"'      Stop frequency :3GHz
14 PRINT #1, "MKZF 5ØØMHZ"' Marker center  :5ØØMHz
15 PRINT #1, "TS"'           Take a sweep
16 '
17 PRINT #1, "MKMHRM"'       Multi marker On & Perform harmonics function
18 '
19 FOR I = 1 TO 5
20 PRINT #1, "MKMP? " + STR$(I)
21 INPUT #1, FREQ'           Input marker frequency data
22 PRINT #1, "MKML? " + STR$(I)
23 INPUT #1, LEVEL'          Input marker frequency data
24 '
25 PRINT USING "Marker No. ## #,###.####MHz ####.##dBm"; I; FREQ / 1ØØØØØØ;
LEVEL
26 NEXT I
27 '
28 END
```

This program allows the frequency to be set using the START-STOP at lines 12 and 13. The "MKZF_500MHZ" at line 14 moves the zone marker center to 500 MHz so that marker can capture a fundamental. (In the initial state, the zone is positioned in the center of the screen. The "MKMHRM" at line 17 sets the multimarker to HARMONICS mode (harmonic frequency measurement).

Respective frequencies and levels at five markers can be read out by setting the number of loops to 5 in the FOR...NEXT statement from line 19 to line 26. The other parts of this program are the same as <Example 5-1>.

Gate functions

<Example 6> Reads out spectrum data by observing the burst wave using the gate function.

```
 1 '+++++++++++++++++++++++++++++++++++++
 2 ' MS2660 series Sample program
 3 '  <<Gate sweep>>
 4 '+++++++++++++++++++++++++++++++++++++
 5 '
 6 ' Setup parameter of PC Com. port
 7 '
 8 OPEN "COM1:24ØØ,N,8,1,CD5ØØ,DSØ,LF" FOR RANDOM AS #1
1Ø '
11 PRINT #1, "INI"'          Initialize Spectrum Analyzer
12 '
13 DIM TRACE(5Ø1)'                 Define read data area
14 PRINT #1, "CF 5ØØMHZ"'          Center fequency :5ØØMHz
15 PRINT #1, "SP 1ØMHZ"'           Span frequency  :1ØMHz
16 PRINT #1, "RB 1ØØKHZ"'          Resolution BW   :1ØØkHz
17 PRINT #1, "TRGSOURCE WIDEVID"'  Trigger source  :Wide IF video
18 PRINT #1, "GD 5ØUS"'            Gate delay      :5Ø usec
19 PRINT #1, "GL 4ØØUS"'           Gate length     :4ØØ usec
2Ø PRINT #1, "GE INT"'             Gate            :Internal timer
21 PRINT #1, "GATE ON"'            Gate sweep On
22 '
23 FOR TMR = Ø TO 25ØØØ
24 NEXT TMR'                       Wait
25 '
26 FOR I = Ø TO 5ØØ'               Read out & print trace data
27     PRINT #1, "XMA? " + STR$(I) + ",1"
28     INPUT #1, TRACE(I)
29     PRINT USING "###.##dBm"; TRACE(I) / 1ØØ
3Ø NEXT I
31 '
32 END
```

When the burst waveform shown in Fig. 6-1 is observed, the spectrum shown in Fig. 6-2 (a) is output. This function can conveniently be used to observe the spectrum of the ON interval (interval shown by A in Fig. 6-1) in this waveform. This program uses the wide IF video trigger signal as a gate source signal.

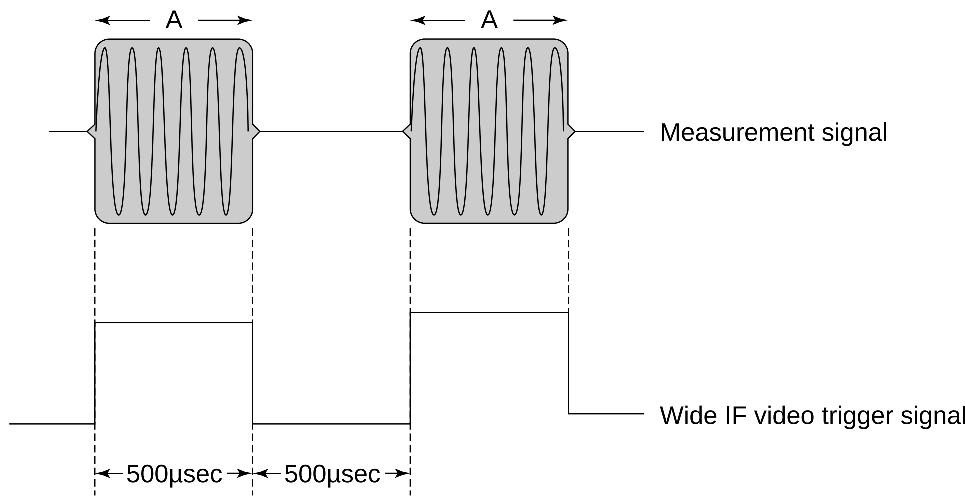


Fig. 6-1 Burst Waveform

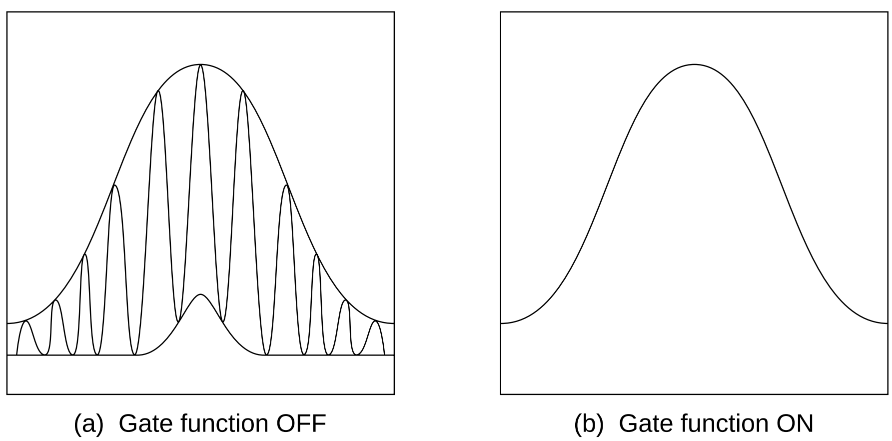


Fig. 6-2 Burst Wave Spectrum

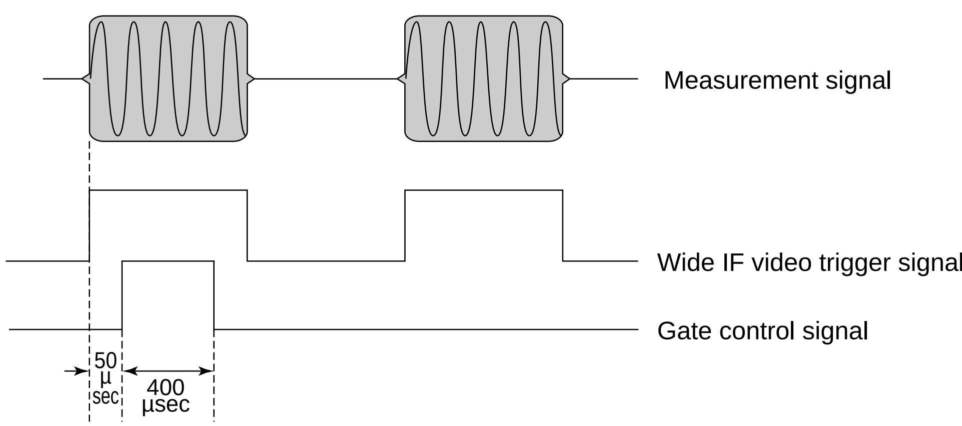


Fig. 6-3 Sample Program for Gate-Control Signal Generation Timing

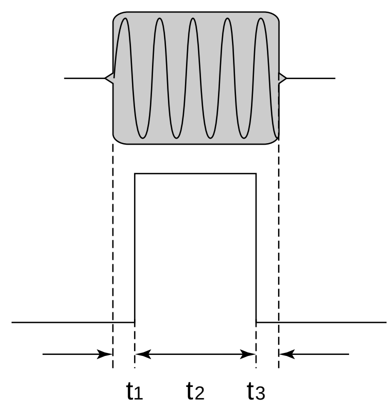
The RBW command at line 16 sets RBW to the optimum value depending on the GATE conditions (GATE DELAY: t1, GATE LENGTH: t2) as shown in Table 6-1 below.

The block from line 17 sets the trigger signal, and the block from lines 18 to 20 sets the gate conditions. The gate function is set to ON at line 21. The waiting time is granted at liens 23 and 24 because it takes time to form a perfect waveform which is fully connected.

The block from liens 26 to 30 allows trace data to be output by the "XMA?" query. The spectrum can be observed as shown in Fig. 6-2(b) by executing this program.

Table 6-1 RBW Optimum Values

RBW	t_1	t_2	t_3
1 kHz	≥3 ms	≥20 µs	≥1 µs
3 kHz	≥1 ms		
10 kHz	≥230 µs		
30 kHz	≥200 µs		
100 kHz	≥20 µs		
300kHz	≥15 µs		
1 MHz 3 MHz	≥10 µs		



(Blank)

Saving and recalling data

<Example 7> Saves and recalls data to and from memory card.

■ Saving data

```
 1 '+++++++++++++++++++++++++++++++++++++++++++++++++++
 2 ' MS2660 series Sample program
 3 '  <<Save parameter & trace data to Memory Card>>
 4 '+++++++++++++++++++++++++++++++++++++++++++++++++++
 5 '
 6 ' Setup parameter of PC Com. port
 7 '
 7 OPEN "COM1:24ØØ,N,8,1,CD5ØØ,DSØ,LF" FOR RANDOM AS #1
 8 '
1Ø GOSUB SAVMEMCARD'           Call Save subroutine
11 '
12 END
13 '
14 '[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[
15  ' SAVE TO MemoryCard SUBROUTINE
16  ']]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
17 SAVMEMCARD:
18  '
19 INPUT "INPUT TITLE"; TTL$'  Enter save file comment(Title)
2Ø PRINT #1, "TITLE '" + TTL$ + "'"
21 '
22 PRINT #1, "PMCS SLOT1"'     Save slot :Slot1(Upper)
23 INPUT "FILE No."; FILE'     Enter save file No.
24 PRINT #1, "SVM" + STR$(FILE)' Perform save procces
25 RETURN
```

■ Recalling data

```
 1 '++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 2 ' MS2660 series Sample program
 3 '  <<Recall parameter &/or trace data from memory card>>
 4 '++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 5 '
 6 ' Setup parameter of PC Com. port
 7'
 8 OPEN "COM1:24Ø Ø ,N,8,1,CD5Ø Ø ,DSØ ,LF" FOR RANDOM AS #1
 9 '
1Ø GOSUB RCLMEMCARD'                         Call recall subroutine
11 END
12 '
13 '[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[
14 ' Recall from memory card SUBROUTINE
15 ']]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
```

```
16 RCLMEMCARD:
17 '
18 PRINT #1, "PMCS SLOT1"'                        Recall slot  :Slot1(Upper)
19 '                                              Enter recall data type
2Ø INPUT "SELECT RECALL DATA  1=TRACE&PARAM  2=PARAM"; RCD
21 IF RCD = 2 THEN RCDATA$ = "P" ELSE RCDATA$ = "TP"
22 PRINT #1, "RDATA " + RCDATA$'                  Set recall data type
23 '
24 INPUT "FILE No."; FILE'                        Enter recall file No.
25 PRINT #1, "RCM" + STR$(FILE)'                  Perform recall procces
26 RETURN
```

These two programs are used as subroutines called from other programs. Each subroutine can be called by placing GOSUB SAVMEMCARD or GOSUB RCLMEMCARD at the line number where the program data is to be saved or restored.

<Example>

```
         .
         .

2ØØ PRINT #1,"SWP"
21Ø GOSUB SAVMEMCARD
         .
         .
         .
```

The block from lines 19 and 20 of SAVMEMCARD sets the title. When the saved data is displayed if the title has been set, this title is also displayed. This can conveniently be used to find data.

The block from lines 22 sets the media to be used for saving to the internal memory card in slot 1 (upper side).

FILE No. is input at line 23 and data is saved to the FILE No. at line 24.

Line 20 of RCLMEMCARD selects the data to be recalled for trace data including parameters or parameters only. Line 22 declares the item to be recalled, and the specified file is recalled at lines 24

Adjacent-channel leakage power measurement

<Example 8> Subroutine for adjacent-channel leakage power measurement

```
'+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
' MS2660 series Sample program
'  <<Adj ch Power measure>>
'+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
'
' Setup parameter of PC Com. port
'
OPEN "COM1:24ØØ,N,8,1,CD5ØØ,DSØ,LF" FOR RANDOM AS #1
'
PRINT #1, "INI"'            Initialize Spectrum Analyzer
'
PRINT #1, "CF 5ØØMHZ"'      Center fequency :5ØØ MHz
PRINT #1, "SP 8ØKHZ"'       Span frequency  :8Ø kHz
'
GOSUB ADJ'                  Call Adj. CH. Power measure subroutine
END
'
'[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[
' Adj ch Power MEASURE SUBROUTINE
']]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
ADJ:
'
PRINT #1, "ADJCH BOTH"
PRINT #1, "ADJCHBW 8.5KHZ"
PRINT #1, "ADJCHSP 12.5KHZ"
PRINT #1, "ADJCHSPF 25KHZ"
PRINT #1, "MADJMOD MOD"
'
PRINT #1, "TS"
PRINT #1, "MEAS ADJ,EXE"
'
PRINT #1, "RES?"'           Query the result
INPUT #1, LWLVL1, UPLVL1, LWLVL2, UPLVL2' Read out the result data
'                           response-1:Lower channel power (near)
'                           response-2:Upper channel power (near)
'                           response-3:Lower channel power (Far)
'                           response-4:Upper channel power (Far)
'
PRINT USING "Lower side CH1 Level=####.###dBm"; LWLVL1
PRINT USING "Upper side CH1 Level=####.###dBm"; UPLVL1
PRINT USING "Lower side CH2 Level=####.###dBm"; LWLVL2
PRINT USING "Upper side CH3 Level=####.###dBm"; UPLVL2
'
RETURN
```

This ADJ program is a subroutine, which requires the center frequency and frequency span to be set to appropriate values in the main program. Then it is executed.

The block from lines 23 to 26 sets adjacent-channel measurement conditions, which is both the upper and lower channels, the 8.5 kHz channel width, 12.5 kHz channel 1 separation , and 25.0 kHz channel 2 separation. After the sweep is executed by the "TS" command at line 29, the adjacent-channel leakage power is measured at line 30. Line 32 queries reading the measured value at line 33.

The program in <Example 8> for measuring a modulated wave relative to the total power can be changed to a program for measurement relative to the reference level by rewriting line 27 as shown below:

```
PRINT #1, "MADJMOD UNMD"
```

In this case, perform the following operations before activating this subroutine.

Put the input signal in the unmodulated state and execute PEAK -> CF and PEAK -> REF. Then return to the modulated state.

Occupied frequency bandwidth measurement

<Example 9> Subroutine for occupied frequency bandwidth measurement using N% of POWER method

```
 1 '+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 2 ' MS2660 series Sample program
 3 '  <<Occ BW measure>>
 4 '+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 5 '
 6 ' Setup parameter of PC Com. port
 7 '
 8 OPEN "COM1:24ØØ,N,8,1,CD5ØØ,DSØ,LF" FOR RANDOM AS #1
 9 '
1Ø PRINT #1, "INI"'          Initialize Spectrum Analyzer
11 '
12 PRINT #1, "CF 5ØØMHZ"'   Center fequency :5ØØMHz
13 PRINT #1, "SP 5ØKHZ"'    Span frequency  :5ØkHz
14 '
15 GOSUB OBW'               Call Occ BW measure subroutine
16 END
17 '
18 '[[[[[[[[[[[[[[[[[[[[[[[[[
19' OBW MEASURE SUBROUTINE
2Ø ']]]]]]]]]]]]]]]]]]]]]]]]]
21 OBW:
22 '
23 PRINT #1, "MOBW N"'      OccBW measure method : n% method
24 PRINT #1, "OBWN 99"'     n%                   : 99%
25 PRINT #1, "DET SMP"'     Detection mode       : Sample
26 PRINT #1, "VAVG 16"'     Average sweep count  : 16
27 PRINT #1, "VAVG ON"'     Average sweep On
28 '
29 PRINT #1, "TSAVG"'       Take average sweep
1Ø '
31 PRINT #1, "MEAS OBW,EXE"'Perform OccBW measure
32 '
33 PRINT #1, "RES?"'        Query the result
34 INPUT #1, OBWFREQ, CNTRFRQ'   Read out the result data
35 '                         response-1:Occ BW frequency
37 '                        response-2:Signal center frequency
38 '
39 PRINT USING "CENTER FREQ=####.###MHz"; CNTRFRQ / 1ØØØØØØ!
4Ø PRINT USING "##%BW  FREQ=####.###kHz"; NPC; OBWFREQ / 1ØØØ
41 '
42 RETURN
```

Line 24 sets the N% value to set n = 99% in <Example 9> by sending the OBWN command for setting the occupied frequency bandwidth to Spectrum Analyzer at line 23 and 24. Line 25 sets the detection mode to SAMPLE. Line 26 set the averaging count and line 27 averaging to ON respectively.

Line 29 issues the "TSAVG command to repeat the sweep by the required number of times for averaging processing. Line 31 measures the occupied frequency bandwidth of the averaging-processed waveform. Line 33 queries reading the occupied frequency bandwidth and the center frequency of the frequency bandwidth at line 34.

To make a measurement using X dB DOWN, rewrite lines 23 and 24 as shown below:

```
            .
            .
PRINT @SPA;"OBWXDB 25"
PRINT @SPA;"MOBW XDB"
            .
            .
```

Setting template data

<Example 10> Subroutine for template data

```
'+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
' MS2660 series Sample program
'  <<Makeup template>>
'+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
'
' Setup parameter of PC Com. port
'
OPEN "COM1:24ØØ,N,8,1,CD5ØØ,DSØ,LF" FOR RANDOM AS #1
'
GOSUB MAKETM'   Call makeup template subroutine
END
'
'[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[
' makeup template SUBROUTINE
']]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
MAKETM:
'
PRINT #1, "MTEMP 1"' Select template No. 1 for making template
PRINT #1, "MTEMPREL ABS"' Set template level to "Absolute"
PRINT #1, "MTEMPINI UP1"' Initialize Limit line-1 upper data
PRINT #1, "MTEMPINI LW1"' Initialize Limit line-1 lower data
'
PRINT #1, "MTEMPL UP1"'   Select Limit line-1 upper for write limit data
RESTORE LMTUP1
'== Limit line-1 upper data ==
LMTUP1:
DATA 8: '  Limit line-1 upper data count
DATA "-5ØUS","-65.2DBM":
DATA "-24US","-65.2DBM":
DATA "-24US","18.8DBM":
DATA "6.643MS","18.8DBM":
DATA "6.643MS","-45.2DBM":
DATA "6.714MS","-45.2DBM":
DATA "6.714MS","-65.2DBM":
DATA "6.95ØMS","-65.2DBM":
'
READ N
FOR I = 1 TO N
' Read each limit data & write to limit line area
  READ TM$, LEV$
  PRINT #1, "MTEMPIN" + STR$(I) + "," + TM$ + "," + LEV$
NEXT I
'
PRINT #1, "MTEMPL LW1"'   Select Limit line-1 lower for write limit data
RESTORE LMTLW1
'== Limit line-1 lower data ==
LMTLW1:
DATA 4: '  Limit line-1 Lower data count
DATA "95US","-2ØØDBM":
DATA "95US","Ø.8DBM":
```

```
52 DATA "6.524MS","Ø.8DBM":
53 DATA "6.524MS","-2ØØDBM":
54 '
55 READ N
56 FOR I = 1 TO N
57 ' Read each limit data & write to limit line area
58   READ TM$, LEV$
59   PRINT #1, "MTEMPIN" + STR$(I) + "," + TM$ + "," + LEV$
6Ø NEXT I
61 '
62 RETURN
63
64
```

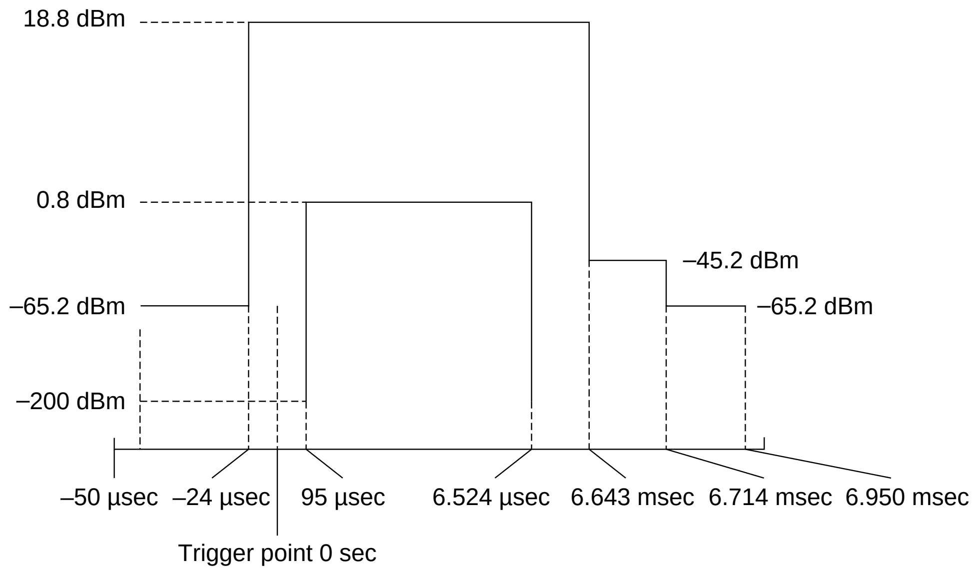


Fig. 6-4 Setting Data

The block from line 18 selects the template No. to be set. The block from line 19 specifies the template data as an absolute value. The block from lines 20 and 21 initializes the current data settings. The block from lines 23 and 37 to 42 sets LIMIT LINE 1 UPPER. Line 23 sets the data to be set in LIMIT LINE1 UPPER. Line 24 specifies the line where setting data is written.

Line 37 reads the number of data points to set the number of loops to N in the FOR ...NEXT statement at lines 38 to 42. Various data settings are read in the FOR...NEXT block.

The block from lines 44 and 54 to 59 sets LIMIT LINE 1 LOWER like the block from lines 23 and 37 to 42.

The block from lines 26 to 35 and 47 to 52 contains the DATA statements for setting the data included in these lines as template data. Lines 26 and 47 are label lines for the RESTORE statement.

Each data item in lines 27 and 48 is numeric, and shows the number of data points. In the DATA statements following the DATA statement with this numeric data, the string expressions are listed as string data with units in order of time and level.

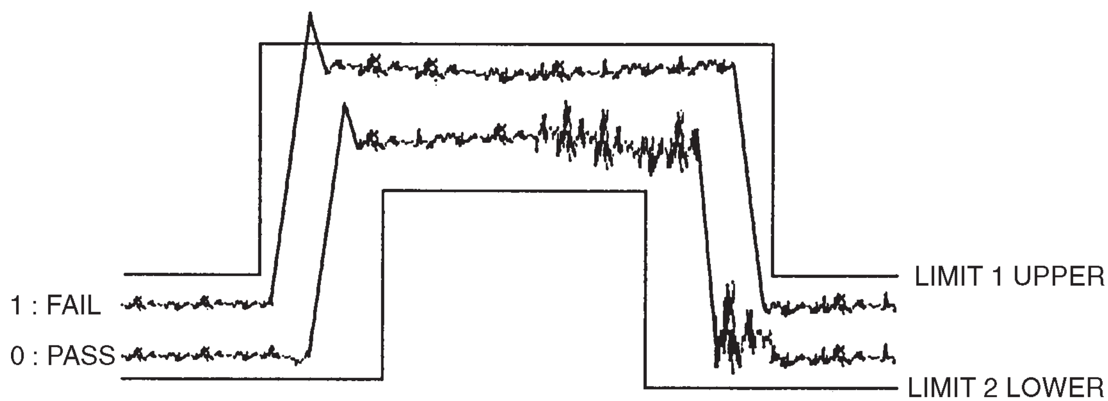
Measuring template

<Example 11> Subroutine for template measurement

```
 1 '+++++++++++++++++++++++++++++++++++++
 2 ' MS2660 series Sample program
 3 '  <<Check template limit>>
 4 '+++++++++++++++++++++++++++++++++++++
 5 '
 6 ' Setup parameter of PC Com. port
 7 '
 8 OPEN "COM1:24ØØ,N,8,1,CD5ØØ,DSØ,LF" FOR RANDOM AS #1
 9 '
1Ø PRINT #1, "INI"'          Initialize Spectrum Analyzer
11 '
12 PRINT #1, "CF 5ØØMHZ"'    Center fequency :5ØØMHz
13 PRINT #1, "DFMT TIME"'    Display         :Trace-Time(Zero span mode)
14 PRINT #1, "TRGSOURCE WIDEVID"'Trigger source  :Wide IF video
15 PRINT #1, "TRGS TRGD"'    Trigger sweep On
16 PRINT #1, "TDY -6ØUS"'    Delay time      :-6Ø usec
17 PRINT #1, "TSP 12MS"'     Time span       :12 msec
18 PRINT #1, "TS"'           Take a sweep
19 '
2Ø GOSUB MEASTMP'            Call template measure subroutine
21 '
22 END
23 '
24 '[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[
25 ' Template measure SUBROUTINE
26 ']]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
27 MEASTMP:
28 '
29 PRINT #1, "TEMP 1"'       Select template 1
3Ø PRINT #1, "TEMPSLCT UP1,ON"'  Limit line-1 upper On
31 PRINT #1, "TEMPSLCT LW1,ON"'  Limit line-1 lower On
32 '
33 PRINT #1, "MEAS TEMP,CHECK"'  Perform template limit check
34 '
35 PRINT #1, "RES?"'         Query the result
36 INPUT #1, CHK1$, CHK2$'   Read out the result
37 '
39 PRINT "LIMIT LINE 1"
4Ø IF CHK1$ = "Ø" THEN
5Ø     PRINT "  CHECK PASS!"
6Ø ELSE
7Ø     PRINT "  CHECK FAIL!"
8Ø END IF
9Ø '
91 RETURN
```

This subroutine checks whether or not a burst signal waveform satisfies the specification using the set template data.

Line 29 specifies the template No. used for a go/no-go decision. Line 30 and 31 specify LIMIT 1 UPPER and LIMIT 1 LOWER as limit lines respectively. Line 33 executes template measurement, line 35 requests data, and line 36 receives data.



When part of a waveform is beyond LIMIT LINE, a response of "1" is generated to indicate FAIL. When the waveform is not beyond LMIT LINE, a response of "0" is generated to indicate PASS.

Burst wave average power measurement

<Example 12> Subroutine for burst wave average power measurement Fig.

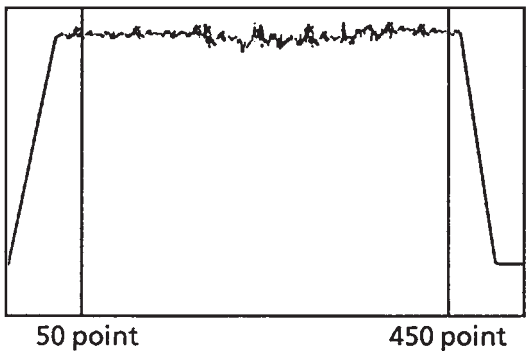
```
 1 '++++++++++++++++++++++++++++++++++++++
 2 ' MS2660 series Sample program
 3 '  <<Burst power measure>>
 4 '++++++++++++++++++++++++++++++++++++++
 5 '
 6 ' Setup parameter of PC Com. port
 7 '
 8 OPEN "COM1:24ØØ,N,8,1,CD5ØØ,DSØ,LF" FOR RANDOM AS #1
 9 '
10 PRINT #1, "INI"'          Initialize Spectrum Analyzer
11 '
12 PRINT #1, "CF 5ØØMHZ"'           Center fequency :5ØØMHz
13 PRINT #1, "DFMT TIME"'           Display         :Trace-Time(Zero span
mode)
14 PRINT #1, "TRGSOURCE WIDEVID"'   Trigger source  :Wide IF video
15 PRINT #1, "TRGS TRGD"'           Trigger sweep On
16 PRINT #1, "TDY -6ØUS"'           Delay time      :-6Ø usec
17 PRINT #1, "TSP 12MS"'                     Time span       :12 msec
18 PRINT #1, "TS"'                  Take a sweep
19 '
20 GOSUB MEASPWR'                   Call burst power measure subroutine
21 '
22 END
23 '
24 '[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[
25 ' Burst power measure SUBROUTINE
26 ']]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
27 MEASPWR:
28 '
29 PRINT #1, "PWRSTART 5Ø"'         Power measure start point :5Ø point(1
div)
30 PRINT #1, "PWRSTOP 45Ø"'         Power measure stop point  :45Ø point(9
div)
31 '
32 PRINT #1, "MEAS POWER,EXE"'       Perform power measure
33 '
34 PRINT #1, "RES?"'                Query the result
35 INPUT #1, PWRDB, PWRW'           Read out the result
36 '
37 PRINT USING "####.##dBm ####.##mW"; PWRDB; PWRW / 1E+Ø9
38 RETURN
```

This program is a subroutine that measures the burst wave average power.

Lines 29 and 30 set the measurement start and stop points on the screen display.

The average power is measured at line 32.

Data can be obtained as a value with dBm units or pW UNITS.



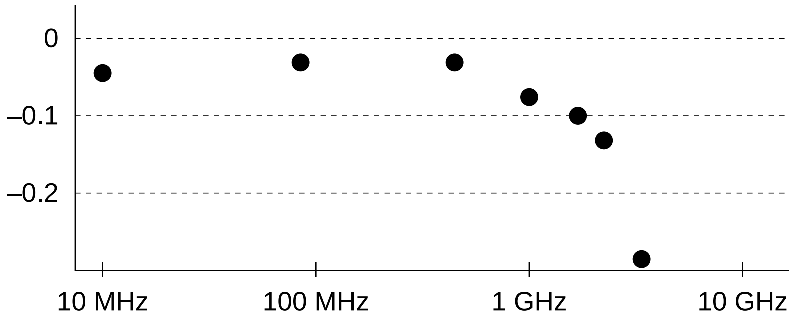
When a waveform is displayed on the screen as shown in the left diagram (TIME domain), the average power between 50 point and 450 point is measured

Before calling the subroutine, lines 12 to 18 set the center frequency, time delay, etc. to execute the sweep.

Frequency characteristic correction data setting

<Example 13>

```
'++++++++++++++++++++++++++++++++++++++++
' MS2660 series Sample program
'  <<Makeup correction factor table>>
'++++++++++++++++++++++++++++++++++++++++
'
' Setup parameter of PC Com. port
'
OPEN "COM1:24ØØ,N,8,1,CD5ØØ,DSØ,LF" FOR RANDOM AS #1
'
GOSUB MAKECORR'           Call makeup correction factor table subroutine
END
'
'[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[
' makeup correction factor table SUBROUTINE
']]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
MAKECORR:
'
PRINT #1, "CORR 1"'       Select template No. 1 for making template
PRINT #1, "CORC"'         Initialize Limit line-1 upper data
'
RESTORE CORRDATA
'== correction factor data ==
CORRDATA:
DATA 7: '                 correction factor data count
DATA "1ØMHZ","-Ø.Ø4DB":
DATA "1ØØMHZ","-Ø.Ø3DB":
DATA "5ØØMHZ","-Ø.Ø3DB":
DATA "1GHZ","-Ø.Ø8DB":
DATA "1.5GHZ","-Ø.1ØDB":
DATA "2GHZ","-Ø.13DB":
DATA "3GHZ","-Ø.29DB":
'
READ N
FOR I = Ø TO N - 1
'                         Read each correction factor data
'                         & write to limit line area
READ FR$, LEV$
PRINT "CORD " + STR$(I) + "," + FR$ + "," + LEV$
PRINT #1, "CORD " + STR$(I) + "," + FR$ + "," + LEV$
 NEXT I
'
RETURN
```



The line 18 selects the correction No. to be set.

The line 19 initializes the correction data being set currently.

The line 21 specifies the line on which data to be set is written.

The lines 25 to 31 specifies the correction data to be set together with the frequency and level data.

The lines 33 to 40 is the frequency characteristic correction data setting section.

The line 33 reads the number of data items to be set. The block from lines 34 to 40 writes the correction data in the loop of the FOR --- NEXT statement. Note that the data No. starts from 0.

When this subroutine MAKECORR executed, the set correction data is written. The frequency correction processing is validated from the subsequent sweep after setting.

Precautions on Creating the GPIB Program

Note the following points when writing remote control programs using GPIB Interface.

No.	Precaution	Description
1	Be sure to initialize each device.	There may be a number of the state in which each device is not proper to be actually used due to operation on its own panel or execution of other programs. It is necessary to using individual devices with a prescribed condition resulting from initializing them. Execute the following. ① Initializing the interface functions (Send IFC) ② Initializing message exchange functions of each device (DevClear) ③ Initializing the functions proper to each device (INI or *RTS)
2	Do not send any command (related to the device) other than the Receive @ statement immediately after sending a query.	If MLA is received when a command other than the Receive @ statement is sent to the controller before the response to a query is read, the output buffer is cleared, and the response message disappears. For this reason, write the Receive @ statement in immediate succession to a query.
3	Create a program that avoids the exception processing of the protocol.	Avoid stoppage of execution (caused by an error) by means of providing a program with exception-processing section against exceptions that can be foreseen.
4	Confirm the interface function of each device (subset).	Execution of program does not advance if necessary subset (s) has (have) not been prepared in the device. Be sure to confirm the subset (s) of each device. Also confirm that each device complies with IEEE488.2.

Initializing (GPIB)

<Example 14> Initializes the MS2660 series.

```
 1 '+++++++++++++++++++++++++++++++++++++++
 2 ' MS2660 series GPIB control sample program
 3 '  <<Initialize GPIB bus & MS2660 Series>>
 4 '+++++++++++++++++++++++++++++++++++++++
 5 REM $INCLUDE: 'C:¥YAT-GPIB¥QBASIC¥QBEDECL.BAS'
 6 DECLARE SUB gpiberr (msg&)
 7 '
 8 SPA% = 1'  Set SPA GPIB adress
 9 CALL SendIFC(Ø)'     Send GPIB bus interface clear
1Ø CALL DevClear(Ø, SPA%)'  Send DeviceClear to MS2660 Series
11 CALL Send(Ø, SPA%, "IP", NLend)'   Send Initialize comand "IP"
12 END
13 '
```

Line 9: Interface-clears GPIB bus.

Line 10: Specifies MS2665C/2667C address, and sends device-clear.

Line 11: Sends "IP" command to for initialization.

There is a '∗RST' command in another GPIB command for executing initialization. The '∗RST' command is used to execute initialization over a winder range. For the range of initialization level, see SECTION 5. The usage of the 'IP' command is identical to the 'INI' command.

For general usage of INI and ∗RST, first initialize the MS2665C/2667C device functions with the IP or INI command, then use the program commands to set only the functions to be changed. This prevents the MS2665C/2667C from being controlled while unnecessary functions are set.

Reading trace data (GPIB)

<Example 15> Performs the same operation as Example 3-1, using GPIB.

```
' +++++++++++++++++++++++++++++++++++++++++++
' MS2660 series GPIB control sample program i
' <<Read out Trace data>>
' +++++++++++++++++++++++++++++++++++++++++++
REM $INCLUDE: 'C : ¥AT-GPIB¥QBASIC¥QBDECL.BAS'
DECLARE SUB gpiberr (msg$)
'
SPA% = 1'                              Set SPA GPIB address
'
'        Initialize GPIB bus & MS2660 Series
CALL SendIFC(Ø)
CALL DevClear(Ø, SPA%)
CALL Send(Ø, SPA%, "IP", NLend)
'
'
CALL Send(Ø, SPA% "CF 5ØØMHZ", NLend)'  Center frequnecy :5ØØMHz
CALL Send(Ø, SPA%, "SP 1ØMHZ", NLend)'  Span frequnecy :1ØMHz
CALL Send(Ø, SPA%, "TS", NLend)         Take a sweep
'
DIM TRACE(5Ø1)'                         Define read data area
CALL Send(Ø, SPA%, "BIN Ø", NLend)'     Set read out data type to
ASCII
'
FOR I = Ø TO 5ØØ'                       Repeat trace(Ø) to
trace(5ØØ):5Ø1 points
CMD$ = "XMA?" + STR$(I) + ",1"
CALL Send(Ø, SPA%, CMD$, NLend)'        Query trace data
'
DATA$ = SPACE$(1ØØ)
CALL Receive(Ø, SPA%, DATA$, NLend)'    Read out trace data
'
TRACE(I) = VAL(DATA$)'                  Store readout data to trace
data area
'                                       Print out trace data
PRINT USING "Trace-A(###) ####.##"; I; TRACE(I)/1ØØ
NEXT I
'
'
END
```

Lines 11 to 13: Initializes GPIB bus and MS2665C/67C/68C.

CALL Send( ) statements after line 13:
Sends MS2665C/67C/68C commands. Command termination code is specified to NLend (line-feed code, New-Line or LF).

CALL Receive( ) statements at line 28:

Reads out trace data from MS2665C/67C/68C.

Termination code of the read data is specified to NLend.

Line 30: Converts the read character-string data to numeric data, and stores it at trace-data store area.

SECTION 7

TABLES OF DEVICE MESSAGES

This section gives information about the device messages of the MS2665C/67C/68C in the form of tables. The messages are arranged according to function, as shown below. For detailed descriptions of commands, see SECTION 8, "DETAILED DESCRIPTIONS OF COMMANDS."

TABLE OF CONTENTS

Table of MS2665C/67C/68C Device Messages (1 /44)

Parameter: Outline	Parameter: Control item	Program command	Query	Response
■Frequency/ Amplitude	**FREQUENCY/ AMPLITUDE**			
• Frequency	**FREQUENCY**			
Selects the mode for setting the frequency band.	FREQ MODE CENTER-SPAN START-STOP	 FRQ△Ø FRQ△2	 FRQ? FRQ?	 FRQ△Ø FRQ△2
Sets the center frequency.	CENTER FREQ	CNF△f CF△f	CNF? CF?	CNF△f f
Steps up the center frequency.	FREQ STEP UP	FUP CF△UP	——— ———	——— ———
Steps down the center frequency.	FREQ STEP DOWN	FDN CF△DN	——— ———	——— ———
Sets the start frequency.	START FREQ	STF△f FA△f	STF? FA?	STF△f f
Sets the stop frequency.	STOP FREQ	SOF△f FB△f	SOF? FB?	SOF△f f
Sets the frequency step size.	FREQ STEP SIZE	FSS△f SS△f	FSS? SS?	FSS△f f
Sets the scroll step size.	SCROLL STEP SIZE 1 div 2 div 5 div 10 div	 SSS△1 SSS△2 SSS△5 SSS△1Ø	 SSS? SSS? SSS? SSS?	 SSS△1 SSS△2 SSS△5 SSS△1Ø
Sets the maximum peak point within BG to the center frequency.	AUTO TUNE	ATUN	———	———
Shifts the spectrum in the left or right direction.	SCROLL LEFT RIGHT	 SCR△Ø SCR△LEFT SCR△1 SCR△RIGHT	 ——— ——— ——— ———	 ——— ——— ——— ———
• Span	**SPAN**			
Sets the frequency span.	FREQ SPAN	SPF△f SP△f	SPF? SP?	SPF△f f

Note:△is a space.

Table of MS2665C/67C/68C Device Messages (2 /44)

Parameter		Program command	Query	Response
Outline	Control item			
■Frequency/ Amplitude	**FREQUENCY AMPLITUDE**			
• Span	**SPAN**			
Steps up the frequency span.	FREQ SPAN STEP UP	SPU SP△UP	— —	— —
Steps down the frequency span.	FREQ SPAN STEP DOWN	SPD SP△DN	— —	— —
Sets to full span.	FULL SPAN	FS	—	—
Sets to zero span.	ZERO SPAN	SPF△Ø	SPF?	SPF△Ø
Select the band (MS2665C/67C)	BAND SELECT AUTO: 0 Hz to 21.2 GHz (67C: 0 Hz to 30.0 GHz)	BNDC△AUTO BND△Ø HNLOCK△OFF HNUNLK	BNDC? BND? HNLOCK? —	AUTO BND△Ø OFF —
	0: 0 Hz to 3.2 GHz	BNDC△Ø BND△1 HNLOCK△Ø HN△Ø	BNDC? BND? HNLOCK? HN?	Ø BND△1 ON Ø
	1–: 2.92 GHz to 6.5 GHz (67C: 3.1 GHz to 6.5 GHz)	BNDC△1- BND△2 HNLOCK△1 HN△1	BNDC? BND? HNLOCK? HN?	1- BND△2 ON 1
	1+: 6.4 GHz to 8.1 GHz	BNDC△1+ BND△3 HNLOCK△2 HN△2	BNDC? BND? HNLOCK? HN?	1+ BND△3 ON 2
	2+: 8.0 GHz to 15.3 GHz	BNDC△2+ BND△4 HNLOCK△3 HN△3	BNDC? BND? HNLOCK? HN?	2+ BND△4 ON 3
	3+: 15.2 GHz to 21.2 GHz (67C: 15 GHz to 22.4 GHz)	BNDC△3+ BND△5 HNLOCK△4 HN△4	BNDC? BND? HNLOCK? HN?	3+ BND△5 ON 4
	4+: 22.3 GHz to 30.0 GHz (MS2667C only)	BNDC△4+ BND△6 HNLOCK△5 HN△5	BNDC? BND? HNLOCK? HN?	4+ BND△6 ON 5

Table of MS2665C/67C/68C Device Messages (3 /44)

Parameter		Program command	Query	Response
Outline	Control item			
Select the band (MS2668C)	BAND SELECT AUTO: 0 Hz to 40 GHz	BNDC△AUTO BND△Ø HNLOCK△OFF HNUNLK	BNDC? BND? HNLOCK? ———	AUTO BND△Ø OFF ———
	0: 0 Hz to 3.2 GHz	BNDC△Ø BND△1 HNLOCK△Ø HN△Ø	BNDC? BND? HNLOCK? HN?	Ø BND△1 ON Ø
	1–: 3.1 GHz to 5.6 GHz	BNDC△1- BND△2 HNLOCK△1 HN△1	BNDC? BND? HNLOCK? HN?	1- BND△2 ON 1
	1+(n=1): 5.4 GHz to 8.1 GHz	BNDC△1+ BND△3 HNLOCK△2 HN△2	BNDC? BND? HNLOCK? HN?	1+ BND△3 ON 2
	1+(n=2): 7.9 GHz to 14.3 GHz	BNDC△1++ BND△4 HNLOCK△3 HN△3	BNDC? BND? HNLOCK? HN?	1++ BND△4 ON 3
	2–(n=4): 14.1 GHz to 26.5 GHz	BNDC△2– BND△5 HNLOCK△4 HN△4	BNDC? BND? HNLOCK? HN?	2– BND△5 ON 4
	3–(n=6): 26.2 GHz to 40 GHz	BNDC△3– BND△6 HNLOCK△5 HN△5	BNDC? BND? HNLOCK? HN?	3– BND△6 ON 5
• Level	**AMPLITUDE**			
Sets the reference level.	REFERENCE LEVEL	RLV△l RL△l	RLV? RL?	RLV△l l
Steps up the reference level.	REF LEVEL STEP UP	LUP RL△UP	———	———
Steps down the reference level.	REF LEVEL STEP DOWN	LDN RL△DN	———	———

Table of MS2665C/67C/68C Device Messages (4 /44)

Parameter: Outline	Parameter: Control item	Program command	Query	Response
■Frequency/ Amplitude	**FREQUENCY/ AMPLITUDE**			
• Level	**AMPLITUDE**			
Sets the LOG scale step size.	LOG SCALE STEP SIZE			
	MANUAL	LSS△l	LSS?	LSS△l
	AUTO			
	1div	LSSA△1	LSSA?	LSSA△1
	2div	LSSA△2	LSSA?	LSSA△2
	5div	LSSA△5	LSSA?	LSSA△5
	10div	LSSA△1Ø	LSSA?	LSSA△1Ø
Sets the LOG scale.	LOG SCALE RANGE			
	1dB/div	SCL△Ø LG△1DB	SCL? LG?	SCL△Ø 1
	2dB/div	SCL△1 LG△2DB	SCL? LG?	SCL△1 2
	5dB/div	SCL△2 LG△5DB	SCL? LG?	SCL△2 5
	10dB/div	SCL△3 LG△1ØDB	SCL? LG?	SCL△3 1Ø
	SCALE UP	LG△UP	———	———
	SCALE DOWN	LG△DN	———	———
Sets the LIN scale.	SCALE LIN RANGE			
	LIN scale switching	LN LG△Ø	——— ———	——— ———
	1%/div	SCL△4	SCL?	SCL△4
	2%/div	SCL△5	SCL?	SCL△5
	5%/div	SCL△6	SCL?	SCL△6
	10%/div	SCL△7	SCL?	SCL△7
Sets the display unit system.	DISPLAY UNIT			
	dBm	UNT△Ø AUNITS△DBM KSA	UNT? AUNITS? ———	UNT△Ø DBM ———
	dBμV	UNT△1 AUNITS△DBUV KSC	UNT? AUNITS? ———	UNT△1 DBUV ———
	dBmV	UNT△2 AUNITS△DBMV KSB	UNT? AUNITS? ———	UNT△2 DBMV ———
	V	UNT△3 AUNITS△V KSD	UNT? AUNITS? ———	UNT△3 V ———
	dBμV(emf)	UNT△4 AUNITS△DBUVE	UNT? AUNITS?	UNT△4 DBUVE
	W	UNT△5 AUNITS△W	UNT? AUNITS?	UNT△5 W

Table of MS2665C/67C/68C Device Messages (5 /44)

Parameter: Outline	Parameter: Control item	Program command	Query	Response
■Frequency/ Amplitude	**FREQUENCY/ AMPLITUDE**			
• Display line	**DISPLAY LINE**			
Sets the Display line ON/OFF.	DISPLAY LINE			
	OFF	DL△OFF	DL?	OFF
	ON	DL△ON		
Sets the Display line level.	DISPLAY LINE LEVEL	DL△l	DL?	l
Marker level/ waveform data Absolute/relative display line	ABS/REL			
	ABS	DSPLV△ABS	DSPLV?	ABS
	REL	DSPLV△REL	DSPLV?	REL
	TRACE-A ABS	DSPLVM△TRA,ABS	DSPLVM?△TRA	ABS
	REL	DSPLVM△TRA,REL	DSPLVM?△TRA	REL
	TRACE-B ABS	DSPLVM△TRB,ABS	DSPLVM?△TRB	ABS
	REL	DSPLVM△TRB,REL	DSPLVM?△TRB	REL
	TRACE-TIME ABS	DSPLVM△TRTIME,ABS	DSPLVM?△TRTIME	ABS
	REL	DSPLVM△TRTIME,REL	DSPLVM?△TRTIME	REL
	TRACE-BG ABS	DSPLVM△TRBG,ABS	DSPLVM?△TRBG	ABS
	REL	DSPLVM△TRBG,REL	DSPLVM?△TRBG	REL
• Reference level offset	**REFERENCE LEVEL OFFSET**			
Offset Offset value	OFFSET			
	OFF	ROFFSET△OFF LVO△Ø	ROFFSET?	OFF
	ON	ROFFSET△ON LVO△1	ROFFSET?	l
	OFFSET VALUE	ROFFSET△l LOS△l	ROFFSET? LOS?	l LOS△l

Table of MS2665C/67C/68C Device Messages (6 /44)

Parameter: Outline	Parameter: Control item	Program command	Query	Response
■Frequency/ Amplitude	**FREQUENCY/ AMPLITUDE**			
• Correction factor relevant	**CORRECTION**			
Selects the type of correction factor.	CORRECTION FACTOR SELECT			
	OFF	CORR△OFF	———	———
		CORR△Ø	CORR?	CORR△Ø
		CDT△Ø	CDT?	CDT△Ø
	ON	CORR△ON	———	———
		CDT△1	CDT?	CDT△1
	CORR1	CORR△1	CORR?	CORR△1
	CORR2	CORR△2	CORR?	CORR△2
	CORR3	CORR△3	CORR?	CORR△3
	CORR4	CORR△4	CORR?	CORR△4
	CORR5	CORR△5	CORR?	CORR△5
Registers the correction factor.	CORRECTION FACTOR† ENTRY	CORD△n,f,l	CORD△n	CORD△f,l
Registers the correction factor name.	CORRECTION FACTOR† LABEL ENTRY	CORRLABEL△n, "text"	CORRLABEL?△n	"text"
Initializes the correction factor.	CORRECTION FACTOR† INITIALIZATION	CORC	———	———
Selects the input impedance.	INPUT IMPEDANCE			
	50 Ω	INZ△5Ø	INZ?	5Ø
	75 Ω	INZ△75	INZ?	75
75 Ω impedance transformer. (MA1621A)	IMPEDANCE TRANSFORMER			
	ON	INPTRNS△ON	INPTRNS?	ON
	OFF	INPTRNS△OFF	INPTRNS?	OFF

† Manual setting is unavailable because the commands are used only for GP-IB.

Table of MS2665C/67C/68C Device Messages (7 /44)

Parameter		Program command	Query	Response
Outline	Control item			
■Display function	**DISPLAY**			
• Display mode	**DISPLAY FUNCTION**			
Selects the display format.	DISPLAY FORMAT			
	TRACE-A	DFMT△A	DFMT?	A
	TRACE-B	DFMT△B	DFMT?	B
	TRACE-TIME	DFMT△TIME	DFMT?	TIME
	TRACE-A/B(A&B)	DFMT△AB1	DFMT?	AB1
	TRACE-A/B(A>B)	DFMT△AB2	DFMT?	AB2
	TRACE-A/B(A<B)	DFMT△AB3	DFMT?	AB3
	TRACE-A/BG (BG>A)	DFMT△ABG1	DFMT?	ABG1
	TRACE-A/BG (BG<A)	DFMT△ABG2	DFMT?	ABG2
	TRACE-A/TIME (TIME>A)	DFMT△ATIME1	DFMT?	ATIME1
	TRACE-A/TIME (TIME<A)	DFMT△ATIME2	DFMT?	ATIME2
• Waveform writing	**WRITE SWITCH**			
Controls writing of the waveform to trace A.	TRACE-A WRITE SWITCH			
	VEIW	AWR△Ø	——	——
		AWR△OFF	AWR?	AWR△OFF
		VIEW△TRA	——	——
	WRITE	AWR△1	——	——
		AWR△ON	AWR?	AWR△ON
		CLRW△TRA	——	——
		A1	——	——
Controls writing of the waveform to trace B.	TRACE-B WRITE SWITCH			
	VIEW	BWR△Ø	——	——
		BWR△OFF	BWR?	BWR△OFF
		VIEW△TRB	——	——
	WRITE	BWR△1	——	——
		BWR△ON	BWR?	BWR△ON
		CLRW△TRB	——	——
		B1	——	——

Table of MS2665C/67C/68C Device Messages (8/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Display function	DISPLAY			
• Waveform writing	DISPLAY FUNCTION			
Controls writing of the waveform to trace BG.	TRACE-BG WRITE SWITCH			
	VIEW	BGWR△Ø	———	———
		BGWR△OFF	BGWR?	BGWR△OFF
		VIEW△TRBG	———	———
	WRITE	BGWR△1	———	———
		BGWR△ON	BGWR?	BGWR△ON
		CLRW△TRBG	———	———
Controls writing of the waveform to trace TIME.	TRACE-TIME WRITE SWITCH			
	VIEW	TMWR△Ø	———	———
		TMWR△OFF	TMWR?	TMWR△OFF
		VIEW△TRTIME	———	———
	WRITE	TMWR△1	———	———
		TMWR△ON	TMWR?	TMWR△ON
		CLRW△TRTIME	———	———
• Storage mode	STORAGE MODE			
Selects the mode for processing the trace A waveform.	TRACE MODE(A)			
	NORMAL	AMD△Ø	AMD?	AMD△Ø
	MAX HOLD	AMD△1	AMD?	AMD△1
		MXMH△TRA	———	———
		A2	———	
	AVERAGE	AMD△2	AMD?	AMD△2
	MIN HOLD	AMD△3	AMD?	AMD△3
	CUMULATIVE	AMD△4	AMD?	AMD△4
	OVER WRITE	AMD△5	AMD?	AMD△5

Table of MS2665C/67C/68C Device Messages (9 /44)

Parameter: Outline	Parameter: Control item	Program command	Query	Response
■Display function	**DISPLAY**			
• Storage mode	**STORAGE MODE**			
Selects the mode for processing the trace B waveform.	TRACE MODE(B)			
	NORMAL	BMD△Ø	BMD?	BMD△Ø
	MAX HOLD	BMD△1	BMD?	BMD△1
		MXMH△TRB	———	———
		B2	———	———
	AVERAGE	BMD△2	BMD?	BMD△2
	MIN HOLD	BMD△3	BMD?	BMDℓ3
	CUMULATIVE	BMD△4	BMD?	BMD△4
	OVER WRITE	BMD△5	BMD?	BMD△5
Selects the mode for processing the trace TIME waveform.	TRACE MODE(TIME)			
	NORMAL	TMMD△Ø	TMMD?	TMMD△Ø
	MAX HOLD	TMMD△1	TMMD?	TMMD△1
	AVERAGE	TMMD△2	TMMD?	TMMD△2
	MIN HOLD	TMMD△3	TMMD?	TMMD△3
	CUMULATIVE	TMMD△4	TMMD?	TMMD△4
	OVER WRITE	TMMD△5	TMMD?	TMMD△5
Average processing	AVERAGE			
	OFF	VAVG△Ø	———	———
		VAVG△OFF	———	———
		KSH	———	———
	ON	VAVG△1	———	———
		VAVG△ON	———	———
		KSG	———	———
Number of trace averaged	NUMBER of TRACE AVERAGE			
	4	AVR△Ø	AVR?	AVR△Ø
	8	AVR△1	AVR?	AVR△1
	16	AVR△2	AVR?	AVR△2
	32	AVR△3	AVR?	AVR△3
	128	AVR△4	AVR?	AVR△4
	n	VAVG△n	VAVG?	n
Average sweep stop mode	AVERAGE SWEEP MODE			
	CONTINUOUS	AVGPAUSE△OFF	AVGPAUSE?	OFF
	PAUSE	AVGPAUSE△ON	AVGPAUSE?	ON

Table of MS2665C/67C/68C Device Messages (10/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Display function	**DISPLAY**			
• Storage mode (Cont)	**STORAGE MODE**			
Hold control stop mode	HOLD SWEEP MODE			
	CONTINUOUS	HOLDPAUSE△Ø	HOLDPAUSE?	Ø
	PAUSE (Times specified)	HOLDPAUSE△n	HOLDPAUSE?	n
Selects detection mode	DETECTION MODE			
	POS PEAK	DET△Ø	——	——
		DET△POS	DET?	POS
	SAMPLE	DET△1	——	——
		DET△SMP	DET?	SMP
	MEG PEAK	DET△2	——	——
		DET△NEG	DET?	NEG
	NORMAL	DET△3	——	——
		DET△NRM	DET?	NRM
Selects detection mode	TRACE-A DETECTION MODE			
	POS PEAK	DETM△TRA,POS	DETM?△TRA	POS
	SAMPLE	DETM△TRA,SMP	DETM?△TRA	SMP
	NEG PEAK	DETM△TRA,NEG	DETM?△TRA	NEG
	NORMAL	DETM△TRA,NRM	DETM?△TRA	NRM
	TRACE-B DETECTION MODE			
	POS PEAK	DETM△TRB,POS	DETM?△TRB	POS
	SAMPLE	DETM△TRB,SMP	DETM?△TRB	SMP
	NEG PEAK	DETM△TRB,NEG	DETM?△TRB	NEG
	NORMAL	DETM△TRB,NRM	DETM?△TRB	NRM
	TRACE-TIME DETECTION MODE			
	POS PEAK	DETM△TRTIME,POS	DETM?△TRTIME	POS
	SAMPLE	DETM△TRTIME,SMP	DETM?△TRTIME	SMP
	NEG PEAK	DETM△TRTIME,NEG	DETM?△TRTIME	NEG
	NORMAL	DETM△TRTIME,NRM	DETM?△TRTIME	NRM

Table of MS2665C/67C/68C Device Messages (11/44)

Parameter: Outline	Parameter: Control item	Program command	Query	Response
■Display function	DISPLAY			
• Time	TIME			
Sets the time delay in the time axis sweep mode.	DELAY TIME	TDLY△t DLT△t	TDLY? DLT?	t DLT△t
Sets the time span in the time axis sweep mode.	TIME SPAN	TSP△t	TSP?	t
Sets the time expand mode ON/OFF.	EXPAND ZONE			
	OFF	TZONE△Ø TZONE△OFF	——— TZONE?	——— OFF
	ON	TZONE△1 TZONE△ON	——— TZONE?	——— ON
Sets the time expand mode ON/OFF.	EXPAND			
	OFF	TEXPAND△Ø TEXPAND△OFF	——— TEXPAND?	——— OFF
	ON	TEXPAND△1 TEXPAND△ON	——— TEXPAND?	——— ON
Sets the start time of the expansion.	ZONE START	TZSTART△t TZSTARTP△p	TZSTART? TZSTARTP?	t p
Sets the magnified range of time expansion.	ZONE SPAN	TZSP△t TZSPP△t	TZSP? TZSPP?	t p
• A/B				
Active marker Trace	ACTIVE MARKER TRACE			
	TRACE A	MKTRACE△TRA	MKTRACE?	TRA
	TRACE B	MKTRACE△TRB	MKTRACE?	TRB
■Trace move/ calculation	TRACE MOVE/CALC			
• Trace move	TRACE MOVE			
Moves trace A to B.	A→B	ATB MOV△TRA,TRB	——— ———	——— ———

Table of MS2665C/67C/68C Device Messages (12/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Trace move/ calculation	TRACE MOVE/CALC			
• Trace move (Cont)	TRACE MOVE			
Moves trace B to A.	B→A	BTA	——	——
		MOV△TRB,TRA	——	——
Replaces trace A by B.	A↔B	AXB	——	——
		EX	——	——
		XCH△TRA,TRB	——	——
		XCH△TRB,TRA	——	——
• Trace calculation	TRACE CALC			
A-B→A	A-B→A			
	OFF	AMB△Ø	——	——
		AMB△OFF	AMB?	OFF
		C1	——	——
	ON	AMB△1	——	——
		AMB△ON	AMB?	ON
		C2	——	——
Calculates A - B.	REFERENCE LINE			
	TOP	RLN△Ø	RLN?	RLN△Ø
	MIDDLE	RLN△1	RLN?	RLN△1
	BOTTOM	RLN△2	RLN?	RLN△2
A+B→A	A+B→A	APB	——	——
NORMALIZE (A-B+DL→A)	NORMALIZE (A-B+DL→A)			
	OFF	AMBPL△Ø	——	——
		AMBPL△OFF	AMBPL?	OFF
	ON	AMBPL△1	——	——
		AMBPL△ON	AMBPL?	ON
■Signal search	SIGNAL SEARCH			
Sets the maximum peak point to the center frequency.	PEAK to CF	PCF	——	——
Sets the maximum peak point to the REF level.	PEAK to REF	PRL	——	——

Table of MS2665C/67C/68C Device Messages (13/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Marker function	**MARKER**			
Selects the marker mode.	MARKER MODE			
	MORMAL	MKR△Ø	MKR?	MKR△Ø
		M2	———	———
	DELTA	MKR△1	MKR?	MKR△1
		MKD	———	———
		M3	———	———
	OFF	MKR△2	MKR?	MKR△2
		MKOFF	———	———
		MKOFF△ALL	———	———
		M1	———	———
Specifies the zone marker center position as a point.	ZONE POSITION (point)	MKZ△p	MKZ?	MKZ△p
		MKP△p	MKP?	p
Specifies the zone marker center position as a frequency or time.	ZONE POSITION (freq or time)			
	FREQ SET	MKZF△f	MKZF?	f
		MKN△f	MKN?	f
	UP	MKN△UP	———	———
	DOWN	MKN△DN	———	———
	TIME SET	MKZF△t	MKZF?	t
		MKN△t	MKN?	t
	UP	MKN△UP	———	———
	DOWN	MKN△DN	———	———
Specifies the zone marker width as a point.	ZONE WIDTH(point)	MZW△p	MZW?	MZW△p
Specifies the zone marker width as a frequency.	ZONE WIDTH(freq)	MZWF△f	MZWF?	f
Specifies the zone marker width as a division.	ZONE WIDTH(div)			
	SPOT	MKW△1	MKW?	MKW△1
	0.5 div	MKW△Ø	MKW?	MKW△Ø
	1 div	MKW△5	MKW?	MKW△5
	2 div	MKW△6	MKW?	MKW△6
	5 div	MKW△7	MKW?	MKW△7
	10 div	MKW△2	MKW?	MKW△2
Marker search mode	MARKER SEARCH MODE			
	PEAK MARKER	MKSRCH△PEAK	MKSRCH?	PEAK
	DIP MARKER	MKSRCH△DIP	MKSRCH?	DIP

Table of MS2665C/67C/68C Device Messages (14/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Marker function	MARKER			
• Marker function (Cont)	MARKER FUNCTION			
Moves the marker frequency to the center frequency.	MKR to CF	MKR△3 MKCF E2	― ― ―	― ― ―
Sets the level at the marker point to the REF level.	MKR to REF	MKR△4 MKRL E4	― ― ―	― ― ―
Sets the marker frequency to the CF step.	MKR to CFstep	MKR△5 MKSS E3	― ― ―	― ― ―
Sets the delta marker frequency to the span.	△MKR to SPAN	MKR△6 MKSP KSO	― ― ―	― ― ―
Sets the zone frequency to the span.	ZONE to SPAN	MKR△7	―	―
• Multimarker	MULTI MARKER			
Multimarker	MULTI MARKER OFF ON	 MKMULTI△Ø MKMULTI△OFF MLO MKMULTI△1 MKMULTI△ON	 ― MKMULTI? ― ― MKMULTI?	 ― OFF ― ― ON
Multimarker mode	MULTI MARKER MODE Registers multimarkers on the peak point in descending order from the maximum level down to the tenth. Registers multimarkers on the harmonic frequency ranging from the reference multimarker frequency up to the tenth.	 MKMHI MHI MKMHRM MHM	 ― ―	 ― ―
Selects the multimarker.	SELECT MULTI MARKER nth marker: Sets to OFF. Sets to ON.	 MKSLCT△n,Ø MKSLCT△n,OFF MSE△n,Ø MKSLCT△n,1 MKSLCT△n,ON MSE△n,1	 ― MKSLCT?△n MSE? ― MKSLCT?△n MSE?	 ― OFF MSE△Ø ― ON MSE△1

Table of MS2665C/67C/68C Device Messages (15/44)

Parameter: Outline	Parameter: Control item	Program command	Query	Response
■Marker function (Cont) • Multimarker	**MARKER** **MULTI MARKER**			
Selects the active marker of the multimarkers.	ACTIVE MARKER	MKACT△n MAC△n	MKACT? MAC?	n MAC△n
Specifies the frequency of the designated multimarker number.	MARKER POSITION	MKMP△n,f MPS△n,p	MKMP?△n MPS?△n	f MPS△p
Clears all registered multimarkers.	CLEAR MULTI MARKER	MKMCL MCL	———	———
Multimarker list	MULTI MARKER LIST OFF ON	 MKLIST△Ø MKLIST△OFF MLI△Ø MKLIST△1 MKLIST△ON MLI△1	 ——— MKLIST? MLI? ——— MKLIST? MLI?	 ——— OFF MLI△Ø ——— ON MLI△1
Multimarker list Sets the level data by distinguishing the absolute value from the relative value.	MULTI MARKER LIST LEVEL ABSOLUTE RELATIVE	MKLLVL△ABS MKLLVL△REL	MKLLVL? MKLLVL?	ABS REL
Multimarker list Sets the frequency data by distinguishing the relative value from the absolute value.	MULTI MARKER LIST FREQUENCY ABSOLUTE RELATIVE	MKLFREQ△ABS MKLFREQ△REL	MKLFREQ? MKLFREQ?	ABS REL
Reads the multimarker level.	MULTI MARKER LEVEL QUERY	———	MKML?△n MLR?△n	l l
Reads the multimarker frequency.	MULTI MARKER FREQUENCY QUERY	———	MFR?△n	f
Reads the multimarker all level/frequency.	MULTI MARKER ALL REVEL/FREQ QUERY	———	MKMFL?	f_1,l_1,f_2,l_2

Table of MS2665C/67C/68C Device Messages (16/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Marker function (Cont)	**MARKER**			
• Peak search	**PEAK SEARCH**			
Peak search mode	PEAK SEARCH MODE	MKS△Ø	——	——
	PEAK	MKPK	——	——
		MKPK△HI	——	——
		E1	——	——
	NEXT PEAK	MKS△1	——	——
		MKPK△NH	——	——
	DIP	MKS△2	——	——
		MKMIN	——	——
	NEXT RIGHT PEAK	MKS△9	——	——
		MKPK△NR	——	——
	NEXT LEFT PEAK	MKS△1Ø	——	——
		MKPK△NL	——	——
	NEXT DIP	MKS△11	——	——
Search resolution	SEARCH RESOLUTION	MKPX△l	MKPX?	l
Search threshold value	SEARCH THRESHOLD			
	OFF	SRCHTH△Ø	——	——
		SRCHTH△OFF	SRCHTH?	OFF
	ON	SRCHTH△1	——	——
		SRCHTH△ON	——	——
	ABOVE	SRCHTH△ABOVE	SRCHTH?	ABOVE
	BELOW	SRCHTH△BELOW	SRCHTH?	BELOW
• Input position	**INPUT POSITION**			
Reads the reference marker position.	REFERENCE MARKER POSITION	——	RMK?	RMK△p
Reads the current marker position.	CURRENT MARKER POSITION	——	CMK?	CMK△p
Reads the frequency at the marker point.	MARKER FREQ QUERY			
	FREQ	——	MKF?	f
	TIME	——	MKF?	t
Reads the level at the marker point.	MARKER LEVEL	——	MKL?	l
		——	MKA?	l

Table of MS2665C/67C/68C Device Messages (17/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Coupled function	**COUPLED FUNCTION**			
Sets the resolution bandwidth.	RESOLUTION BANDWIDTH			
	MANUAL	ARB△Ø	ARB?	ARB△Ø
	AUTO	ARB△1	ARB?	ARB△1
		RB△AUTO	———	———
		CR	———	———
	10 Hz	RB△1ØHZ	RB?	1Ø
	(Option)	RBW△13	RBW?	RBW△13
	30 Hz	RB△3ØHZ	RB?	3Ø
	(Option)	RBW△Ø	RBW?	RBW△Ø
	100 Hz	RB△1ØØHZ	RB?	1ØØ
	(Option)	RBW△1	RBW?	RBW△1
	300 Hz	RB△3ØØHZ	RB?	3ØØ
	(Option)	RBW△2	RBW?	RBW△2
	1 kHz	RB△1KHZ	RB?	1ØØØ
		RBW△3	RBW?	RBW△3
	3 kHz	RB△3KHZ	RB?	3ØØØ
		RBW△4	RBW?	RBW△4
	10 kHz	RB△1ØKHZ	RB?	1ØØØØ
		RBW△5	RBW?	RBW△5
	30 kHz	RB△3ØKHZ	RB?	3ØØØØ
		RBW△6	RBW?	RBW△6
	100 kHz	RB△1ØØKHZ	RB?	1ØØØØØ
		RBW△7	RBW?	RBW△7
	300 kHz	RB△3ØØKHZ	RB?	3ØØØØØ
		RBW△8	RBW?	RBW△8
	1 MHz	RB△1MHZ	RB?	1ØØØØØØ
		RBW△9	RBW?	RBW△9
	3 MHz	RB△1MHZ	RB?	3ØØØØØØ
		RBW△14	RBW?	RBW△14
	RBW UP	RB△UP	———	———
	RBW DOWN	RB△DN	———	———

Table of MS2665C/67C/68C Device Messages (18/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Coupled function	**COUPLED FUNCTION**			
Sets the video bandwidth.	VIDEO BANDWIDTH			
	MANUAL	AVB△Ø	AVB?	AVB△Ø
	AUTO	AVB△1	AVB?	AVB△1
		VB△AUTO	———	———
		CV	———	———
	1 Hz	VB△1HZ	VB?	1
		VBW△Ø	VBW?	VBW△Ø
	3 Hz	VB△3ØHZ	VB?	3
		VBW△8	VBW?	VBW△8
	10 Hz	VB△1HZ	VB?	1Ø
		VBW△1	VBW?	VBW△1
	30 Hz	VB△3ØHZ	VB?	3Ø
		VBW△9	VBW?	VBW△9
	100 Hz	VB△1ØØHZ	VB?	1ØØ
		VBW△2	VBW?	VBW△2
	300 Hz	VB△3ØØHZ	VB?	3ØØ
		VBW△1Ø	VBW?	VBW△1Ø
	1 kHz	VB△1KHZ	VB?	1ØØØ
		VBW△3	VBW?	VBW△3
	3 kHz	VB△3KHZ	VB?	3ØØØ
		VBW△11	VBW?	VBW△11
	10 kHz	VB△1ØKHZ	VB?	1ØØØØ
		VBW△4	VBW?	VBW△4
	30 kHz	VB△3ØKHZ	VB?	3ØØØØ
		VBW△12	VBW?	VBW△12
	100 kHz	VB△1ØØKHZ	VB?	1ØØØØØ
		VBW△5	VBW?	VBW△5
	300 kHz	VB△3ØØKHZ	VB?	3ØØØØØ
		VBW△13	VBW?	VBW△13
	1 MHz	VB△1MHZ	VB?	1ØØØØØØ
		VBW△7	VBW?	VBW△7
	3 MHz	VB△3MHZ	VB?	3ØØØØØØ
		VBW△14	VBW?	VBW△14
	OFF	VB△OFF	VB?	OFF
		VBW△6	VBW?	VBW△6
		AVB△2	AVB?	AVB△2
	VBW UP	VB△UP	———	———
	VBW DOWN	VB△DN	———	———
Sets the VBW/RBW ratio (where VBW = AUTO).	VBW/RBW RATIO RATIO=r	VBR△r	VBR?	r

Table of MS2665C/67C/68C Device Messages (19/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Coupled function (Cont)	**COUPLED FUNCTION**			
Sets the RBW/Span ON/OFF (Where RBW=AUTO).	RBW/Span			
	OFF	RBSPAN△OFF RBSPAN△Ø	RBSPAN?	OFF
	ON	RBSPAN△ON RBSPAN△1	RBSPAN?	ON
Sets the RBW/Span Ratio.	RBW/Span RATIO	RBR△r	RBR?	r
Sets the sweep time.	SWEEP TIME			
	MANUAL	AST△Ø	AST?	AST△Ø
	AUTO	AST△1 STØ CT	AST? ——— ———	AST△1 ——— ———
	SWEEP TIME SET			
	TIME=t	SWT△t ST△t	SWT? ST?	SWT△t t
	UP	ST△UP	———	———
	DOWN	ST△DN	———	———
Sets the RF attenuator.	RF ATTENUATOR			
	MANUAL	AAT△Ø	AAT?	AAT△Ø
	AUTO	AAT△1 AT△AUTO CA	AAT? ——— ———	AAT△1 ——— ———
Sets the RF attenuator.	0 dB	ATT△Ø AT△Ø	ATT? AT?	ATTØ Ø
	10 dB	ATT△1 AT△1Ø	ATT? AT?	ATT△1 1Ø
	20 dB	ATT△2 AT△2Ø	ATT? AT?	ATT△2 2Ø
	30 dB	ATT△3 AT△3Ø	ATT? AT?	ATT△3 3Ø
	40 dB	ATT△4 AT△4Ø	ATT? AT?	ATT△4 4Ø
	50 dB	ATT△5 AT△5Ø	ATT? AT?	ATT△5 5Ø
	60 dB	ATT△12 AT△6Ø	ATT? AT?	ATT△12 6Ø
	70 dB	ATT△13 AT△7Ø	ATT? AT?	ATT△13 7Ø
	UP	AT△UP	———	———
	DOWN	AT△DN	———	———

Table of MS2665C/67C/68C Device Messages (20/44)

Parameter: Outline	Parameter: Control item	Program command	Query	Response
■Coupled function (Cont)	**COUPLED FUNCTION**			
Sets the bandwidth/ sweep time to AUTO mode.	RBW,VBW/SWEEP TIME AUTO	BSAUTO	——	——
Sets the coupled function to AUTO mode.	COUPLED FUNCTION AUTO	AUTO	——	——
Sets the coupled function at the frequency domain/ time domain.	COUPLE MODE			
	COMMON	VBCOUPLE△COM	VBCOUPLE?	COM
	INDEPENDENCE	VBCOUPLE△IND	VBCOUPLE?	IND
■Sweep function	**SWEEP CONTROL**			
Sets the zone sweep ON/OFF.	ZONE SWEEP			
	OFF	PSW△Ø	——	——
		PSW△OFF	PSW?	PSW△OFF
	ON	PSW△1	——	——
		PSW△ON	PSW?	PSW△ON
Sets the tracking function.	TRACKING			
	OFF	MKTRACK△Ø	——	——
		MKTRACK△OFF	MKTRACK?	OFF
		MTØ	——	——
	ON	MKTRACK△1	——	——
		MKTRACK△ON	MKTRACK?	ON
		MT1	——	——
Sets the sweep mode to single.	SINGKE SWEEP MODE	SNGLS	——	——
		S2	——	——
Executes/checks single sweep.	SINGLE SWEEP/ SWEEP STATUS			
	Executing single sweep	SWP	——	——
		TS	——	——
	Checking the sweep status			
	Sweep completed	——	SWP?	SWP△Ø
	Sweep in progress	——	SWP?	SWP△1
Executes average sweep.	TAKE AVERAGE SWEEP	TSAVG	——	——
Executes hold sweep.	TAKE HOLD SWEEP	TSHOLD	——	——

Table of MS2665C/67C/68C Device Messages (21/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Sweep function	SWEEP CONTROL			
Continuous sweep mode.	COTINUOUS SWEEP MODE	CONTS S1	——— ———	——— ———
Stops the sweep.	SWEEP STOP	SWSTOP	———	———
Restarts the sweep.	SWEEP RESTART	SWSTART	———	———
■Save/Recall	SAVE/RECALL			
Recalls data from the internal memory.	RECALL DATA FROM INTERNAL MEMORY	RGRC△r RC△r	——— ———	——— ———
Recalls data from the memory card.	RECALL DATA FROM MEMORY CARD	RCM△r	———	———
Recalls data from the memory card. Changes the storage mode to View.	WRITE OFF RECALL DATA	RCS△r	———	———
Saves data in the internal memory.	SAVE DATA INTO INTERNAL MEMORY	RGSV△s SV△s	——— ———	——— ———
Saves data on the memory card.	SAVE DATA INTO MEMORY CARD	SVM△s	———	———
Sets the recall data	RECALLED DATA TRACE&PARAM PARAM ONLY TRACE&PARAM(VIEW) PARAM(EXCEPT REF LEVEL)	 RDATA△TP RDATA△P RDATA△TPV RDATA△PER	 RDATA? RDATA? RDATA? RDATA?	 TP P TPV PER
Saves by BMP format	SAVE BMP FILE	SVBMP△n	———	———
■Hard copy	HARD COPY			
Direct plot	DIRECT PLOT START DIRECT PLOT	PLS△Ø PLOT PRINT	——— ——— ———	——— ——— ———

Table of MS2665C/67C/68C Device Messages (22/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Hard copy (cont)	**HARD COPY**			
• Controls hard copy.	**COPY CONTROL**			
Direct plotting device selection.	DIRECT PLOT DEVICE			
Selects the plotter.	PLOTTER			
	HP-GL	PMOD△Ø	PMOD?	PMOD△Ø
	GP-GL	PMOD△1	PMOD?	PMOD△1
	BMP FORMAT	PMOD△4	PMOD?	PMOD△4
Selects the printer.	PRINTER			
	VP-600(ESC/P)	PMOD△2	PMOD?	PMOD△2
	HP-2225	PMOD△3	PMOD?	PMOD△3
Print magnification	PRINT MAGNIFICATION			
	1X1	PRINTMAG△11	PRINTMAG?	11
	2X1	PRINTMAG△21	PRINTMAG?	21
	1X2	PRINTMAG△12	PRINTMAG?	12
	2X2	PRINTMAG△22	PRINTMAG?	22
	2X3	PRINTMAG△23	PRINTMAG?	23
	2X4	PRINTMAG△24	PRINTMAG?	24
Sets the printer GP-IB address.	PRINTER ADDRESS SET	PRIA△a	PRIA?	a
Sets the plotter GP-IB address.	PLOTTER ADDRESS SET	PLTA△a	PLTA?	a
Sets the size of paper output from the plotter.	DIRECT PLOT SIZE			
	A4	PLF△Ø	PLF?	PLF△Ø
	A3	PLF△1	PLF?	PLF△1
Sets the size of the plot.	PLOT AREA			
	FULL SIZE	PLTARA△FULL	PLTARA?	FULL
	QUATER SIZE	PLTARA△QTR	PLTARA?	QTR
Sets the location of the plot on the paper.	PLOT LOCATION			
	Renewed automatically	PLTLC△AUTO	PLTLC?	AUTO
	Fixed at upper left-corner	PLTLC△UPLEFT	PLTLC?	UPLEFT
	Fixed at upper right-corner	PLTLC△UPRIGHT	PLTLC?	UPRIGHT
	Fixed at lower left-corner	PLTLC△LOWLEFT	PLTLC?	LOWLEFT
	Fixed at lower right-corner	PLTLC△LOWRIGHT	PLTLC?	LOWRIGHT
Sets the size of the plot.	PRINTER PORT			
	RS232C	PRTPORT△RS232C	PRTPORT?	RS232C
	GPIB	PRTPORT△GPIB	PRTPORT?	GPIB
	PARALLEL	PRTPORT△PARALLEL	PRTPORT?	PARALLEL
	NONE	PRTPORT△NONE	PRTPORT?	NONE

Table of MS2665C/67C/68C Device Messages (23/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Hard copy (cont)	HARD COPY			
• Controls hard copy.	COPY CONTROL			
Selects the item(s) to be output to the plotter.	DIRECT PLOT OUTPUT ITEM			
	ALL	PLI△Ø	PLI?	PLI△Ø
	TRACE ONLY	PLI△1	PLI?	PLI△1
	SCALE ONLY	PLI△2	PLI?	PLI△2
Selects "UPPER LEFT" for the plot location on the paper (only in AUTO ADVANCE mode).	PLOTTER LOCATION PRESET	PLTHOME	———	———
■Measure function	MEASURE			
Sets the measure function to OFF.	MEASURE FUNCTION ALL OFF	MEAS△OFF	MEAS?	OFF
• Noise measurement	NOISE MEASURE			
Measures the noise.	NOISE MEASURE			
	OFF	MEAS△NOISE,OFF	———	———
	ON	MEAS△NOISE,ON	MEAS?	NOISE
	ABSOLUTE executed	MEAS△NOISE,ABS	MEAS?	NOISE
	C/N RATIO executed	MEAS△NOISE,CN	MEAS?	CN
	Transferring measured results (dBm/ch or dBm/Hz)	———	RES?	l
Calculation method	ABSOLUTE	MNOISE△ABS	MNOISE?	ABS
	C/N RATIO	MNOISE△CN	MNOISE?	CN
• Occupied frequency bandwidth measurement	OBW MEASURE			
Measures the occupied frequency bandwidth.	OBW MEASURE			
	Executes calculation.	MEAS△OBW,EXE	MEAS?	OBW
	Executes(X dB DOWN).	MEAS△OBW,XDB	MEAS?	OBW
	Executes (N%).	MEAS△OBW,N	MEAS?	OBW
	Transferring measured results (f1: Occupied bandwidth f2: Center frequency)	———	RES?	f1,f2
Calculation method	X dB DOWN method	MOBW△XDB	MOBW?	XDB
	N% method	MOBW△N	MOBW?	N
Sets the conditions of occupied frequency bandwidth.	OBW VALUE			
	x dB	OBWXDB△XDB	OBWXDB?	x
	n%	OBWN△n	OBWN?	n

Table of MS2665C/67C/68C Device Messages (24/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Measure function (Cont)	**MEASURE**			
• Adjacent channel measurement	**ADJACENT CH MEASURE**			
Measures the adjacent channel.	ADJACENT CH MEASURE			
	Executes calculation.	MEAS△ADJ,EXE	MEAS?	ADJ
	Executes (UNMODULATED CARRIER).	MEAS△ADJ,UNMD	MEAS?	ADJ
	Executes(MODULATED CARRIER)	MEAS△ADJ,MOD	MEAS?	ADJ
	Executes(INBAND)	MEAS△ADJ,INABAND	MEAS?	ADJ
	Transferring measured results ($l_{L}1$: CH1 lower sideband, $l_{U}1$: CH1 upper sideband, $l_{L}2$: CH2 lower sideband, $l_{U}2$: CH2 upper sideband)	———	RES?	$l_{L}1$, $l_{U}1$, $l_{L}2$, $l_{U}2$
Selects the adjacent channel.	ADJACENT CH SELECT			
	BOTH SIDES	ADJCH△BOTH	ADJCH?	BOTH
	UPPER SIDE	ADJCH△UP	ADJCH?	UP
	LOWER SIDE	ADJCH△LOW	ADJCH?	LOW
	OFF	ADJCH△OFF	ADJCH?	OFF
Sets the adjacent channel bandwidth.	ADJACENT CH BANDWIDTH	ADJCHBW△f	ADJCHBW?	f
Sets adjacent channel 1 separation.	ADJACENT CH1 SEPARATION	ADJCHSP△f	ADJCHSP?	f
Sets adjacent channel 2 separation.	ADJACENT CH2 SEPARATION	ADJCHSPF△f	ADJCHSPF?	f
Selects the calculation method.	R:TOTAL POWER(MOD)	MADJMOD△MOD	MADJMOD?	MOD
	R:REF LEVEL (UNMOD)	MADJMOD△UNMD	MADJMOD?	UNMD
	R:INBAND	MADJMOD△INABAND	MADJMOD?	INBAND
Sets the graph display ON/OFF.	GRAPH			
	OFF	MADJGRAPH△OFF	MADJGRAPH?	OFF
	ON	MADJGRAPH△ON	MADJGRAPH?	ON
Inband ch Bandwidth Setting	INBAND:CH BANDWIDTH	ADJINBW△f	ADJINBW?	f

Table of MS2665C/67C/68C Device Messages (25/44)

Parameter: Outline	Parameter: Control item	Program command	Query	Response
■Measure function	**MEASURE**			
• Adjacent channel measurement (Cont)	**ADJACENT CH MEASURE**			
Sets the channel center line display ON/OFF.	CHANNEL CENTER LINE			
	OFF	MADJCTRLN△OFF	MADJCTRLN?	OFF
	ON	MADJCTRLN△ON	MADJCTRLN?	ON
Sets the channel range line display ON/OFF.	CHANNEL BAND LINE			
	OFF	MADJBWLN△OFF	MADJBWLN?	OFF
	ON	MADJBWLN△ON	MADJBWLN?	ON
Sets the Inband ch range line display ON/OFF.	INBAND CHANNEL BAND LINE			
	OFF	MADJINBWLN△OFF	MADJINBWLN?	OFF
	ON	MADJINBWLN△ON	MADJINBWLN?	ON
• Template measurement	**TEMPLATE**			
Measures the template.	TEMPLATE MEASURE			
	OFF	MEAS△TEMP,OFF	———	———
	ON	MEAS△TEMP,ON	———	———
	CHECK TEMP	MEAS△TEMP,CHECK	MEAS?	TEMP
	Transferring measured results (c1:LIMIT1 check result (c2:LIMIT2 check result)	———	RES?	c1,c2 (PASS=Ø, FAIL=1)
Moves the template.	TEMPLATE MOVE			
	MOVE X	TEMPMVX△t	TEMPMVX?	t
	MOVE Y	TEMPMVY△l	TEMPMVY?	l
	SAVE	TEMPMSV	———	———
	CANCEL	TEMPMCL	———	———
Selects the template.	SELECT TEMPLATE No.			
	1	TEMP△1	TEMP?	1
	2	TEMP△2	TEMP?	2
	3	TEMP△3	TEMP?	3
	4	TEMP△4	TEMP?	4
	5	TEMP△5	TEMP?	5

Table of MS2665C/67C/68C Device Messages (26/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Measure function	**MEASURE**			
• Template measurement (Cont)	**TEMPLATE**			
Selects the LIMIT line.	SELECT LIMIT LINE			
	LIMIT1 UPPER			
	OFF	TEMPSLCT△UP1,Ø TEMPSLCT△UP1,OFF	――― TEMPSLCT?UP1	――― OFF
	ON	TEMPSLCT△UP1,1 TEMPSLCT△UP1,ON	――― TEMPSLCT?UP1	――― ON
	LIMIT2 UPPER			
	OFF	TEMPSLCT△UP2,Ø TEMPSLCT△UP2,OFF	――― TEMPSLCT?UP2	――― OFF
	ON	TEMPSLCT△UP2,1 TEMPSLCT△UP2,ON	――― TEMPSLCT?UP2	――― ON
	LIMIT1 LOWER			
	OFF	TEMPSLCT△LW1,Ø TEMPSLCT△LW1,OFF	――― TEMPSLCT?LW1	――― OFF
	ON	TEMPSLCT△LW1,1 TEMPSLCT△LW1,ON	――― TEMPSLCT?LW1	――― ON
	LIMIT2 LOWER			
	OFF	TMPSLCT△LW2,Ø TMPSLCT△LW2,OFF	――― TEMPSLCT?LW2	――― OFF
	ON	TMPSLCT△LW2,1 TMPSLCT△LW2,ON	――― TEMPSLCT?LW2	――― ON
• Power measurement	**POWER MEASURE**			
Measures the power.	POWER MEASURE			
	MEASURE Transferring measured results (l:dBm value w: pW value)	MEAS△POWER,EXE ―――	MEAS? RES?	POWER l,w
Sets the point where power measurement starts.	POWER MEASURE START	PWRSTART△p	PWRSTART?	p
Sets the point where power measurement ends.	POWER MEASURE STOP	PWRSTOP△p	PWRSTOP?	p

Table of MS2665C/67C/68C Device Messages (27/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Measure function (Cont)	MEASURE			
• Mask measurement	MASK			
Measures the mask.	MASK MEASURE			
	OFF	MEAS△MASK,OFF	———	———
	ON	MEAS△MASK,ON	———	———
	CHECK TEMP	MEAS△MASK,CHECK	MEAS?	MASK
	Result input c_1:LIMIT1 Check result c_2:LIMIT2 Check result	———	RES?	C1,C2 (PASS=Ø FAIL=1)
Moves the mask.	MASK MOVE			
	MOVE X	MASKMVX△f	MASKMVX?	f
	MOVE Y	MASKMVY△l	MASKMVY?	l
	SAVE	MASKMSV	———	———
	CANCEL	MASKMCL	———	———
Selects the mask.	SELECT MASK No.			
	1	MASK△1	MASK?	1
	2	MASK△2	MASK?	2
	3	MASK△3	MASK?	3
	4	MASK△4	MASK?	4
	5	MASK△5	MASK?	5

Table of MS2665C/67C/68C Device Messages (28/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Measure function	MEASURE			
• Mask measurement (Cont)	MASK			
Selects the LIMIT line.	SELECT LIMIT LINE			
	LIMIT1 UPPER			
	OFF	MASKSLCT△UP1,Ø	———	———
		MASKSLCT△UP1,OFF	MASKSLCT?UP1	OFF
	ON	MASKSLCT△UP1,1	———	———
		MASKSLCT△UP1,ON	MASKSLCT?UP1	ON
	LIMIT2 UPPER			
	OFF	MASKSLCT△UP2,Ø	———	———
		MASKSLCT△UP2,OFF	MASKSLCT?UP2	OFF
	ON	MASKSLCT△UP2,1	———	———
		MASKSLCT△UP2,ON	MASKSLCT?UP2	ON
	LIMIT1 LOWER			
	OFF	MASKSLCT△LW1,Ø	———	———
		MASKSLCT△LW1,OFF	MASKSLCT?LW1	OFF
	ON	MASKSLCT△LW1,1	———	———
		MASKSLCT△LW1,ON	MASKSLCT?LW1	ON
	LIMIT2 LOWER			
	OFF	MASKSLCT△LW2,Ø	———	———
		MASKSLCT△LW2,OFF	MASKSLCT?LW2	OFF
	ON	MASKSLCT△LW2,1	———	———
		MASKSLCT△LW2,ON	MASKSLCT?LW2	ON
• Template management function	MANAGE TEMPLATE			
Selects the template number.	SELECT TEMPLATE No.			
	1	MTEMP△1	MTEMP?	1
	2	MTEMP△2	MTEMP?	2
	3	MTEMP△3	MTEMP?	3
	4	MTEMP△4	MTEMP?	4
	5	MTEMP△5	MTEMP?	5
Selects the LIMIT line.	SELECT LIMIT LINE			
	LIMIT1 UPPER	MTEMPL△UP1	MTEMPL?	UP1
	LIMIT2 UPPER	MTEMPL△UP2	MTEMPL?	UP2
	LIMIT1 LOWER	MTEMPL△LW1	MTEMPL?	LW1
	LIMIT2 LOWER	MTEMPL△LW2	MTEMPL?	LW2

Table of MS2665C/67C/68C Device Messages (29/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Measure function	MEASURE			
• Template management function (Cont)	MANAGE TEMPLATE			
	TEMPLATE LEVEL MODE			
Sets the level data by distinguishing the relative value from the absolute value.	ABSOLUTE	MTEMPREL△OFF	MTEMPREL?	OFF
	RELATIVE	MTEMPREL△ON	MTEMPREL?	ON
Adds 1 point to template data.	INSERT TEMPLATE POINT DATA	MTEMPIN△p,t,l	——	——
Changes 1 point of template data.	REPLACE TEMPLATE POINT DATA	MTEMPRP△p,t,l	——	——
Reads 1 point of template data.	READ TEMPLATE POINT DATA	——	MTEMPPD?△p	t,l
Deletes 1 point of template data.	TEMPLATE POINT DATA DELETE	MTEMPDEL△p	——	——
Initializes the template data.	INITIATE LINE/TEMPLATE			
	LIMIT1 UPPER	MTEMPINI△UP1	——	——
	LIMIT2 UPPER	MTEMPINI△UP2	——	——
	LIMIT1 LOWER	MTEMPINI△LW1	——	——
	LIMIT2 LOWER	MTEMPINI△ LW2	——	——

Table of MS2665C/67C/68C Device Messages (30/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Measure function	MEASURE			
• Template management function (Cont)	MANAGE TEMPLATE			
Specifies how the template data is displayed.	DISPLAY TEMPLATE MODE			
	GRAPH	MTEMPDSP△GRAPH	MTEMPDSP?	GRAPH
	LIST	MTEMPDSP△LIST	MTEMPDSP?	LIST
Sets the template label.	TEMP LABEL	MTEMPLABEL△n, 'text'	MTEMPLABEL?n	text
• Mask management function	MANAGE MASK			
Selects the mask number.	SELECT MASK No.			
	1	MMASK△1	MMASK?	1
	2	MMASK△2	MMASK?	2
	3	MMASK△3	MMASK?	3
	4	MMASK△4	MMASK?	4
	5	MMASK△5	MMASK?	5
Selects the LIMIT line.	SELECT LIMIT LINE			
	LIMIT1 UPPER	MMASKL△UP1	MMASKL?	UP1
	LIMIT2 UPPER	MMASKL△UP2	MMASKL?	UP2
	LIMIT1 LOWER	MMASKL△LW1	MMASKL?	LW1
	LIMIT2 LOWER	MMASKL△LW2	MMASKL?	LW2
Sets the level data by distinguishing the relative value from the absolute value.	MASK LEVEL MODE			
	ABSOLUTE	MMASKREL△OFF	MMASKREL?	OFF
	RELATIVE	MMASKREL△ON	MMASKREL?	ON
Adds 1 point to mask data.	INSERT MASK POINT DATA	MMASKIN△p,t,l	———	———
Changes 1 point of mask data.	REPLACE MASK POINT DATA	MMASKRP△p,t,l	———	———

Table of MS2665C/67C/68C Device Messages (31/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Measure function	**MEASURE**			
• Mask management function (Cont)	**MANAGE MASK**			
Reads 1 point of mask data.	READ MASK POINT DATA	———	MMASKPD?△p	t,l
Deletes 1 point of mask data.	DELETE MASK POINT DATA	MMASKDEL△p	———	———
Initializes the mask data.	INITIATE LINE/MASK			
	LIMIT1 UPPER	MMASKINI△UP1	———	———
	LIMIT2 UPPER	MMASKINI△UP2	———	———
	LIMIT1 LOWER	MMASKINI△LW1	———	———
	LIMIT2 LOWER	MMASKINI△LW2	———	———
Specifies how the mask data is displayed.	DISPLAY MASK MODE			
	GRAPH	MMASKDSP△GRAPH	MMASKDSP?	GRAPH
	LIST	MMASKDSP△LIST	MMASKDSP?	LIST
Sets the mask label.	MASK LABEL	MMASKLABEL△n, 'text'	MMASKLABEL?n	text
• Channel Power Measure				
Measuring Channel Power	Channel Power Measure			
	ON	MEAS△CHPWR,ON	MEAS?	CHPWR
	OFF	MEAS△CHPWR,OFF	———	———
Correction Factor	Correction Factor	CHPWRFACT△l	CHPWRFACT?	l
■Calibration	**CALIBRATION**			
Executes calibration with the internal CAL signal.	CALIBRATION			
	ALL	CAL△Ø	———	———
	FREQ	CAL△1	———	———
	LEVEL	CAL△2	———	———
	FM	CAL△3	———	———
Sets the frequency calibration function ON/OFF.	FREQ CAL			
	OFF	FCAL1Ø△Ø	FCAL1Ø?	Ø
	ON	FCAL1Ø△1	FCAL1Ø?	1

Table of MS2665C/67C/68C Device Messages (32/44)

Parameter: Outline	Parameter: Control item	Program command	Query	Response
■Calibration	**CALIBRATION**			
	PRESELECTOR TUNE			
	MANUAL	PRESEL△a	PRESEL?	a
	AUTO	PRESEL△AUTO	——	——
		PP	——	——
	PRESET	PRESEL△PRESET	——	——
■RS-232C	**RS-232C**			
Sets the baud rate.	BAUD RATE			
	1200	BAUD△1200	BAUD?	1200
	2400	BAUD△2400	BAUD?	2400
	4800	BAUD△4800	BAUD?	4800
	9600	BAUD△9600	BAUD?	9600
Sets the parity.	PARITY			
	EVEN	PRTY△EVEN	PRTY?	EVEN
	ODD	PRTY△ODD	PRTY?	ODD
	OFF	PRTY△OFF	PRTY?	OFF
Sets the data bit.	DATA BIT			
	7bit	DATB△7	DATB?	7
	8bit	DATB△8	DATB?	8
Sets the stop bit.	STOP BIT			
	1bit	STPB△1	STPB?	1
	2bit	STPB△2	STPB?	2
Sets the period of reception time-out.	TIME OUT	TOUT△t	TOUT?	t
■Title	**TITLE**			
Title entry	TITLE ENTRY	TITLE△'text'	TITLE?	text
		KSE△'text'	——	——
		TEN△x,y,'text'	——	——
Title display	TITLE DISPLAY			
	OFF	TTL△0	——	——
		TTL△OFF	TTL?	TTL△OFF
	ON	TTL△1	——	——
		TTL△ON	TTL?	TTL△ON

Table of MS2665C/67C/68C Device Messages (33/44)

Parameter		Program command	Query	Response
Outline	Control item			
■CAL/UNCAL	CAL/UNCAL			
Couple failure	UNCAL			
	UNCAL DISPLAY			
	OFF	UNC△Ø	——	——
		UNC△OFF	UNC?	UNC△OFF
	ON	UNC△1	——	——
		UNC△ON	UNC?	UNC△ON
	UNCAL STATUS			
	NORMAL	——	UCL?	UCL△Ø
	UNCAL	——	UCL?	UCL△1
■Spectrum data	SPECTRUM DATA			
Trace A memory	TRACE-A MEMORY	XMA△p,b	XMA?△p,b	b
Trace B memory	TRACE-B MEMORY	XMB△p,b	XMB?△p,b	b
Trace BG memory	TRACE-BG MEMORY	XMG△p,b	XMG?△p,b	b
Trace TIME memory	TRACE-TIME MEMORY	XMT△p,b	XMT?△p,b	b
Selects ASCII/ Binary.	ASCII DATA	BIN△Ø	——	——
		——	——	
	BINARY DATA	BIN△1	——	——
		——	——	
■PTA control	PTA CONTROL			
Switches the PTA function ON/OFF.	PTA SWITCH			
	OFF	PTA△OFF	——	——
		PTA△Ø	PTA?	PTA△Ø
	ON	PTA△ON	——	——
		PTA△1	PTA?	PTA△1
Selects the mode for controlling PTA via GP-IB.	PTL I/O MODE			
	OFF	PTL△Ø	——	——
	INPUT(COMMAND PROGRAM)	PTL△1	——	——
	OUTPUT (PROGRAM)	——	PTL?	text

Table of MS2665C/67C/68C Device Messages (34/44)

Parameter		Program command	Query	Response
Outline	Control item			
■PTA control (Cont)	PTA CONTROL			
Writes/reads the dual port memory.	DUAL-PORT MEMORY READ/WRITE	PMY△a,"b"	PMY△a,c	"b"
Selects the control port for GP-IB.	CONTROL PORT SELECT			
	RS-232C	PORT△1	PORT?	PORT△1
	GPIB	PORT△2	PORT?	PORT△2
	PARALLEL (CENTRO)	PORT△3	PORT?	PORT△3
Defines the menu set.	DEFINE MENUSET	MENUSET△n, text,…	——	——
Defines the menu.	DEFINE MENU	MENU△n,text,…	——	——
Opens the menu set.	OPEN MENUSET	MSOPEN△n	——	——
Initializes the contents of the menu definition.	CLEAR MENU DEFINE	CLRMENU	——	——
Displays the entry prompt message.	OPEN ENTRY	ENTRY△text,n,a	——	——
Reads the entry data.	READ ENTRY	——	ENTRY?	a
PTA execution State	PTA STATUS			
	PTA ON	PTA△1		
	PTA OFF	PTA△Ø		
	READY	——	PTA?	PTA△Ø
	BREAK	——	PTA?	PTA△1
	BUSY	——	PTA?	PTA△2
	RUN	——	PTA?	PTA△3
PTL mode	PTL MODE			
	PTL ON	PTL△1	——	——
	PTL OFF	PTL△1	——	——
	READOUT PTL STATEMENT	——	PTL?	(PTL STATEMENT)
Event generation	EVENT			
	DELETE	EDLY△t	——	——
	TIME	ETIM△t1,t2,t3	——	——
	CYCLICAL	ECYC△t	——	——

Table of MS2665C/67C/68C Device Messages (35/44)

Parameter		Program command	Query	Response
Outline	Control item			
■PTL Library	PTA LIBRARY			
Library down load	PTA LIBRARY			
	START DOWN LOAD	DOWNLOAD	——	——
	DOWNLOAD END	LOADEND	——	——
Library file	LIBRARY FILE			
	SAVE	SAVELIB△a[,b,c,···]	——	——
	LOAD	LOADLIB△a	——	——
Common variable	COMMON VARIABLE	VAR△a,b	VAR?△a	:b
Array common variable	COMMON ARRAY			
	DEFINE	DIM△a,b[,c]	——	——
	ARRAY VARIABLE	DVAR△a,b,c,d	DVAR?△a,b[,c]	d
Library execution	EXECUTE LIBRARY	lib△name	——	——
■Others	ETC.			
Terminator	TERMINATOR			
	LF	TRM△Ø	——	——
	CR/LF	TRM△1	——	——
Performs level-3 initialization of measurement control parameters.	INITIALIZE	INI	——	——
		IP	——	——
partial initialization	PARATIAL PRESET			
	PRESET ALL	PINI△Ø	——	——
	PRESET SWEEP CONTLOL	PINI△1	——	——
	PRESET TRACE PARAMETER	PINI△2	——	——
	PRESET LEVEL PARAMETER	PINI△3	——	——
	PRESET FREQ/TIME PARAMETER	PINI△4	——	——
Sets the built-in clock.	TIMER SET			
	DATE	DATE△yyyy,mm,dd	DATE?	yyyy,mm,dd
	TIME	TIME△hh,mm,ss	TIME?	hh,mm,ss
Calculates how long the device has been powered on.	TIME COUNT READ	——	TMCNT?	t(hr)

Table of MS2665C/67C/68C Device Messages (36/44)

Parameter: Outline	Parameter: Control item	Program command	Query	Response
■Others (Cont)	**ETC.**			
LCD display	LCD DISPLAY			
	OFF	DISPLAY△OFF	―――	―――
	ON	DISPLAY△ON	―――	―――
Power-on state	POWER ON STATE			
	FIXED STATE(PRESET)	POWERON△IP	POWERON?	IP
	BEFORE POWER OFF	POWERON△LAST	POWERON?	LAST
	RECALL MEMORY	POWERON△n	POWERON?	n
Erase error message	ERASE ERROR MESSAGE	HOLD	―――	―――
Selects the parameter display type.	PARAMETER DISPLAY TYPE			
	TYPE-1	PARADSP△1	PARADSP?	1
	TYPE-2	PARADSP△2	PARADSP?	2
	TYPE-3	PARADSP△3	PARADSP?	3
Time display	TIME DISPLAY			
	OFF	TIMEDSP△OFF	TIMEDSP?	OFF
	ON	TIMEDSP△ON	TIMEDSP?	ON
Selects the date display mode.	DATE DISPLAY MODE			
	YYYY/MM/DD	DATEMODE△YMD	DATEMODE?	YMD
	DD-MM-YYYY	DATEMODE△DMY	DATEMODE?	DMY
	MMM-DD-YYYY	DATEMODE△MDY	DATEMODE?	MDY
Selects the comment column display type.	COMMENT DISPLAY			
	TITLE	COMMENT△TITLE	COMMENT?	TITLE
	TIME	COMMENT△TIME	COMMENT?	TIME
	OFF	COMMENT△OFF	COMMENT?	OFF
Selects the display color pattern.	COLOR PATTERN			
	PATTERN-1	COLORPTN△COLOR1	COLORPTN?	COLOR1
	PATTERN-2	COLORPTN△COLOR2	COLORPTN?	COLOR2
	PATTERN-3	COLORPTN△COLOR3	COLORPTN?	COLOR3
	PATTERN-4	COLORPTN△COLOR4	COLORPTN?	COLOR4
	USER PATTERN	COLORPTN△USERCOLOR	COLORPTN?	USERCOLOR
Copies the display color pattern to the user pattern.	COPY COLOR PATTERN			
	PATTERN-1	COPYCOLOR△COLOR1	―――	―――
	PATTERN-2	COPYCOLOR△COLOR2	―――	―――
	PATTERN-3	COPYCOLOR△COLOR3	―――	―――
	PATTERN-4	COPYCOLOR△COLOR4	―――	―――

Table of MS2665C/67C/68C Device Messages (37/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Others (Cont)	ETC.			
Defines the user color pattern.	DEFINE USER COLOR n,r,g,b	COLORDEF△	COLORDEF?△n	r,g,b
Reads the error code.	READ OUT ERROR CODE	———	ERROR?	e1,e2
Auto set sweep time	AUTO SWEEP TIME			
	FAST	ASWT△FAST	ASW?	FAST
	NORMAL (HI-LEVEL ACCURCY)	ASWT△SLOW	ASW?	SLOW
Erase Warm up message	ERASE WARM UP MESSAGE	ERASEWUP	———	———
			POWERON?	IP
Execute frequency domain sweep	FREQ DOMAIN SWEEP			
	LOCK BY SWEEP	FRQDOMAIN△LOCK	FRQDOMAIN?	LOCK
	UNLOCK	FRQDOMAIN△UNLOCK	FRQDOMAIN?	UNLOCK
	UNLOCK COUNT	UNLOCKCOUNT△n	UNLOCKCOUNT?	n
Execute zero span sweep mode	ZERO SPAN SWEEP MODE			
	DIGITAL SWEEP	ZEROSPNMODE△DIGITAL	ZEROSPNMODE?	DIGITAL
	ANALOG SWEEP	ZEROSPNMODE△ANALOG	ZEROSPNMODE?	ANALOG
Composite mode	COMPOSITE MODE			
	NORMAL	COMP△NRM	COMP?	NRM
	PAL	COMP△PAL	COMP?	PAL
	NTSC	COMP△NTSC	COMP?	NTSC

Table of MS2665C/67C/68C Device Messages (38/44)

Parameter: Outline	Parameter: Control item	Program command	Query	Response
■Common command and event status	**GPIB COMMON COMMAND:EVENT STATUS**			
Clears the Status Byte Register.	CLEAR STATUS COMMAND	*CLS	———	———
Sets the bit in the Service Request Enable Register.	SERVICE REQUEST ENABLE	*SRE△n	*SRE?	n
Returns the current value of the Status Byte.	READ STATUS BYTE	———	*STB?	n
Executes single sweep.	TRIGGER COMMAND	*TRG	———	———
Executes the self test.	SELF TEST	———	*TST	n
Keeps the next command on standby during execution of a device command.	WAIT TO CONTINUE	*WAI	———	———
Returns the manufacturer name, model name, etc. of the product.	IDENTIFICATION QUERY	———	*IDN?	ANRITSU…
Perform a level-3 device reset.	RESET COMMAND	*RST	———	———
Synchronization mode between device and controller	OPERATION COMPLETE WAITING FOR SERVICE REQUEST WAITING FOR OUTPUT QUEUE IN DEVICE	 *OPC ———	 ——— *OPC?	 ——— 1
Sets or clears the Standard Event Status Enable Register.	STANDARD EVENT ENABLE STATUS	*ESE△n	*ESE?	n
Reads the Standard Event Status Enable Register.	STANDARD EVENT STATUS REGISTER	———	*ESR?	n
Controls masking of the Extended Event Status.	EVENT STATUS ENABLE	ESE2△n	ESE2?	n
Reads the Extended Event Status.	EVENT STATUS REGISTER	———	ESR2?	n

Table of MS2665C/67C/68C Device Messages (39/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Frequency counter	FREQUENCY COUNT			
• Frequency measurement	FREQ MEASURE			
Measures the frequency.	FREQ MEASURE			
	OFF	MKC△Ø	MKC?	MKC△Ø
		MC△OFF	———	———
		MKFC△Ø	MKFC?	Ø
		MKFC△OFF	———	———
		MEAS△FREQ,OFF	———	———
	ON	MKC△1	MKC?	MKC△1
		MC△ON	———	———
		MKFC△1	MKFC?	1
		MKFC△ON	———	———
		MEAS△FREQ,ON	MEAS?	FREQ
	Transferring measured results	———	RES?	f
Sets the counter to the specified resolution.	COUNT RESOLUTION			
	1 Hz	CRS△Ø	CRS?	CRS△Ø
		MKFCR△1HZ	MKFCR?	1
	10 Hz	CRS△1	CRS?	CRS△1
		MKFCR△1ØHZ	MKFCR?	1Ø
	100 Hz	CRS△2	CRS?	CRS△2
		MKFCR△1ØØHZ	MKFCR?	1ØØ
	1 kHz	CRS△3	CRS?	CRS△3
		MKFCR△1KHZ	MKFCR?	1ØØØ
	FREQ UP	MKFCR△UP	———	———
	FREQ DOWN	MKFCR△DN	———	———
■FM demodulation waveform monitor	FM MONITOR			
Sets the function for monitoring the FM demodulation waveform.	FM MONITOR			
	OFF	SPFUNC△OFF	SPFUNC?	OFF
	FM MONITOR MONITOR	SPFUNC△FM	SPFUNC?	FM
Sets the bandwidth for demodulating FM.	FM RANGE	FMRNG△f	FMRNG?	f
Sets the coupling to the FM waveform monitor.	COUPLING			
	AC COUPLING	COUPLE△AC	COUPLE?	AC
	DC COUPLING	COUPLE△DC	COUPLE?	DC

Table of MS2665C/67C/68C Device Messages (40/44)

Parameter		Program command	Query	Response
Outline	Control item			
■<u>Trigger/gate sweep</u>	**<u>TRIGGER/GATE SWEEP</u>**			
Gate function	GATE MODE			
	OFF	GATE△Ø	———	———
		GATE△OFF	GATE?	OFF
		GMD△Ø	GMD?	GMD△Ø
	ON	GATE△1	———	———
		GATE△ON	GATE?	ON
		GMD△1	GMD?	GMD△1
Sets the gate delay time.	GATE DELAY TIME	GD△t	GD?	t
		GDL△t	GDL?	GDL△t
Sets the gate length.	GATE LENGTH	GL△t	GL?	t
		GLN△t	GLN?	GLN△t
Sets internal or external termination of the gate interval.	GATE END			
	INTERNAL	GE△INT	GE?	INT
		GED△Ø	GED?	GED△Ø
	EXTERNAL	GE△EXT	GE?	EXT
		GED△1	GED?	GED△1
Sets the trigger mode (sets the trigger source/trigger switch).	TRIGGER MODE			
	FREERUN	TRG△Ø	TRG?	TRG△Ø
		TM△FREE	TM?	FREE
	VIDEO	TRG△1	TRG?	TRG△1
		TM△VID	TM?	VID
	LINE	TRG△2	TRG?	TRG△2
		TM△LINE	TM?	LINE
	EXT	TRG△3	TRG?	TRG△3
		TM△EXT	TM?	EXT
	WIDE IF VIDEO	TRG△7	TRG?	TRG△7
		TM△WIDEVID	TM?	WIDEVID
Sets the trigger switch.	TRIGGER SWITCH			
	FREERUN	TRGS△FREE	TRGS?	FREE
	TRIGGERD	TRGS△TRGD	TRGS?	TRGD

Table of MS2665C/67C/68C Device Messages (41/44)

Parameter: Outline	Parameter: Control item	Program command	Query	Response
■Sweep function	**SWEEP CONTROL**			
Sets the trigger source.	TRIGGER SOURCE			
	VIDEO	TRGSOURCE△VID	TRGSOURCE?	VID
	LINE	TRGSOURCE△LINE	TRGSOURCE?	LINE
	EXT	TRGSOURCE△EXT	TRGSOURCE?	EXT
	WIDE IF VIDEO	TRGSOURCE△WIDEVID	TRGSOURCE?	WIDEVID
Sets the external trigger level type (when the trigger source = EXT).	EXT TRIGGER TYPE			
	±10 V	EXTTYPE△1ØV	EXTTYPE?	1ØV
	TTL	EXTTYPE△TTL	EXTTYPE?	TTL
Sets the sweep trigger threshold level.	TRIGGER LEVEL	TRGLVL△l	TRGLVL?	l
		TLV△l	TLV?	TLV△l
Selects the sweep trigger slope.	TRIGGER SLOPE			
	RISE	TRGSLP△RISE	TRGSLP?	RISE
		TSL△1	TSL?	TSL△1
	FALL	TRGSLP△FALL	TRGSLP?	FALL
		TSL△Ø	TSL?	TSL△Ø
Sets the time-out period for the trigger sweep wait (this is also the time-out period of the GP-IB talker function).	SWEEP TIME OUT	GTOUT△t	GTOUT?	t

Table of MS2665C/67C/68C Device Messages (42/44)

Parameter		Program command	Query	Response
Outline	Control item			
■AM/FM sound monitor	**AM/FM SOUND MONITOR**			
• Sound	**SOUND**			
Selects the function for monitoring the sound from the detector output.	AM/FM SOUND MONITOR			
	OFF	MON△OFF MAM△Ø MFM△Ø	MON? MAM? MFM?	OFF MAM△Ø MFM△Ø
	AM	MON△AM MAM△1	MON? MAM?	AM MAM△1
	FM	MON△FM MFM△1	MON? MFM?	FM MFM△1
	FM NARROW	MON△FMNARROW	MON?	FMNARROW
Adjusts the volume of the sound monitor.	AM/FM SOUND MONITOR VOLUME	MONVOL△V MVL△v	MONVOL? MVL?	v MVL△v
■GP-IB interface	**GP-IB**			
Sets the time-out period for the GP-IB talker function (this is also the period for the trigger sweep wait time-out).	GPIB TIME OUT	GTOUT△t	GTOUT?	t

Table of MS2665C/67C/68C Device Messages (43/44)

Parameter		Program command	Query	Response
Outline	Control item			
■Memory Card	**MEMORY CARD**			
Selects the Memory Card slot.	SLOT SELECT SLOT1 SLOT2	 PMCS△SLOT1 PMCS△SLOT2	 PMCS? PMCS?	 SLOT1 SLOT2
Saves the template data file.	SAVE TEMPLATE FILE	TEMPSAVE△n	———	———
Loads the template data file.	LOAD TEMPLATE FILE	TEMPLOAD△n	———	———
Saves the mask data file.	SAVE MASK FILE	MASKSAVE△n	———	———
Loads the mask data file.	LOAD MASK FILE	MASKLOAD△n	———	———
Saves the correction data file.	SAVE CORRECTION FILE	CORRSAVE△n	———	———
Loads the correction data file.	LOAD CORRECTION FILE	CORRLOAD△n	———	———
Saves the menu definition data file.	SAVE MENU DEFINE FILE	MENUSAVE△n	———	———
Loads the menu definition data file.	LOAD MENU DEFINE FILE	MENULOAD△n	———	———

Table of MS2665C/67C/68C Device Messages (44/44)

Parameter: Outline	Parameter: Control item	Program command	Query	Response
■<u>External mixer</u> (MS2667C/68C only)	**<u>EXTERNAL MIXER</u>**			
Selects mixer mode	MIXER MODE			
	INTERNAL	MXRMODE△INT	MXRMODE?	INT
	EXTERNAL	MXRMODE△EXT	MXRMODE?	EXT
Mixer bias	MIXER BIAS	MBIAS△n	MBIAS?	n
Conversion loss	CONVERSION LOSS	CNVLOSS△l	CNVLOSS?	l
Select the external mixer band	BAND SELECT			
	K: 18.0 GHz to 26.5 GHz	FULBAND△K	FULBAND?	K
	A: 26.5 GHz to 40.0 GHz	FULBAND△A	FULBAND?	A
	Q: 33.0 GHz to 50.0 GHz	FULBAND△Q	FULBAND?	Q
	U: 40.0 GHz to 60.0 GHz	FULBAND△U	FULBAND?	U
	V: 50.0 GHz to 75.0 GHz	FULBAND△V	FULBAND?	V
	E: 60.0 GHz to 90.0 GHz	FULBAND△E	FULBAND?	E
	W: 75.0 GHz to 110.0 GHz	FULBAND△W	FULBAND?	W
	F: 90.0 GHz to 140.0 GHz	FULBAND△F	FULBAND?	F
	D: 110.0 GHz to 170.0 GHz	FULBAND△D	FULBAND?	D
	G: 140.0 GHz to 220.0 GHz	FULBAND△G	FULBAND?	G
	J: 220.0 GHz to 325.0 GHz	FULBAND△J	FULBAND?	J
Signal Identifier	SIGNAL IDENTIFIER			
	OFF	SIGID△Ø SIGID△OFF	SIGID?	0
	ON	SIGID△1 SIGID△ON	SIGID?	1
■<u>Frequency offset</u> (MS2667C/68C only)				
Offset mode Offset value	FREQUENCY OFFSET MODE			
	OFF	FOFMD△Ø FOFMD△OFF	FOFMD?	0
	ON	FOFMD△1 FOFMD△ON	FOFMD?	1
	OFFSET FREQUENCY	FOFFSET△f	FOFFSET?	f

SECTION 8

DETAILED DESCRIPTION OF COMMANDS

This section describes the usable device and response messages in alphabetic order.

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

TABLE OF CONTENTS (continued)

TABLE OF CONTENTS (continued)

SECTION 8 DETAILED DESCRIPTION OF COMMANDS

This section gives detailed descriptions of the device messages for the MS2665C/67C/68C spectrum analyzer in alphabetical order.

Message name spelled out

Message headline

CNF

CNF Center Frequency

■ Function Sets the center frequency (same function as CF).

Program command message

Program query message

Response message

Header	Program command	Query	Response
CNF	CNF△f	CNF?	CNF△f f=-100000000 to 0 to 3000000000 Transfers the data with no suffix code in units of 1 Hz.

■ Value of f –100MHz to 3.0GHz

■ Suffix code

None:	Hz(10^0)	• The data to the left of the colon is part of the program or response data • The data is to the right of the colon.
HZ:	Hz(10^0)	
KHZ,KZ:	kHz(10^3)	
MHZ,MZ:	MHz(10^6)	
GHZ,GZ:	GHz(10^9)	

■ Initial setting Value of f=1.50 GHz

Device-dependent initial setting value

■ Example

```
CNF△123456
CNF△5ØMHz
CNF?
```

■ Restrictions according to the model type and options

None

A1

A1 Trace A Write ON

■ Function Clears trace A waveform data to set the write mode to ON (same function as AWR△1/CLRW△TRA).

Header	Program command	Query	Response
A1	A1	————	————

■ Example A1

A2

A2 Trace A Max Hold

■ Function Controls writing of the waveform data to trace BG.

Header	Program command	Query	Response
A2	A2	————	————

■ Example A2

AAT

AAT RF Attenuator

■ Function Switches the RF attenuator setting mode to AUTO or MANUAL.

Header	Program command	Query	Response
AAT	AAT△sw	AAT?	AAT△sw

■ Value of sw Ø: MANUAL
1: AUTO
■ Suffix code None
■ Initial setting 1:AUTO
■ Example AAT△1

ADJCH

ADJCH Adjacent CH Select

■ Function Selects the subject channel to be calculated for an adjacent channel.

Header	Program command	Query	Response
ADJCH	ADJCH△a	ADJCH?	a

■ Value of a BOTH: BOTH SIDES
UP: UPPER SIDE
LOW: LOWER SIDE
OFF: OFF
■ Suffix code None
■ Initial setting BOTH: BOTH SIDES
■ Example ADJCH△BOTH
ADJCH△LOW

ADJCHBW

ADJCHBW Adjacent CH Bandwidth

■ Function Sets the bandwidth of the adjacent channel.

Header	Program command	Query	Response
ADJCHBW	ADJCHBW△f	ADJCHBW?	f f=10 to 9999990 Transfers the data with no suffix code in units of 1 Hz.

■ Value of f 10 Hz to 9.99999 MHz (10 Hz resolution. Data below 10 Hz is truncated.)

■ Suffix code
None: Hz(10^0)
HZ: Hz(10^0)
KHZ,KZ: kHz(10^3)
MHZ,MZ: MHz(10^6)
GHZ,GZ: GHz(10^9)

■ Initial setting 8.5KHZ: 8.5kHz

■ Example ADJCHBW△8.5KHZ

ADJCHSP

ADJCHSP Adjacent CH Sepalation

■ Function Sets the separation of adjacent channel 1.

Header	Program command	Query	Response
ADJCHSP	ADJCHSP△f	ADJCHSP?	f f=0 to 9999990 Transfers the data with no suffix code in units of 1 Hz.

■ Value of f 0 Hz to 9.99999 MHz (10 Hz resolution. Data below 10 Hz is truncated.)

■ Suffix code
None: Hz(10^0)
HZ: Hz(10^0)
KHZ,KZ: kHz(10^3)
MHZ,MZ: MHz(10^6)
GHZ,GZ: GHz(10^9)

■ Initial setting 12.5KHZ: 12.5kHz

■ Example ADJCHSP△12.5kHz

ADJCHSPF

ADJCHSPF Adjacent CH2 Separation

■ Function Sets the separation of adjacent channel 2.

Header	Program command	Query	Response
ADJCHSP	ADJCHSPF△f	ADJCHSPF?	f f=0 to 9999990 Transfers the data with no suffix code in unit of 1 Hz.

■ Value of f 0 Hz to 9.99999 MHz (10 Hz resolution. Data below 10 Hz is truncated.)

■ Suffix code
None: Hz(10^0)
HZ: Hz(10^0)
KHZ,KZ: kHz(10^3)
MHZ,MZ: MHz(10^6)
GHZ,GZ: GHz(10^9)

■ Initial setting 12.5KHZ: 12.5kHz

■ Example ADJCHSPF△12.5kHz

ADJINBW

ADJINBW Adjacent Inband CH Bandwidth

■ Function Sets the bandwidth of the adjacent inband channel

Header	Program command	Query	Response
ADJINBW	ADJINBW△f	ADJINBW?	f f=10 to 9999990 Transfers the data with no suffix code in unit of 1 Hz.

■ Value of f 10Hz to 9.99999 MHz (10Hz resolution, Data below 10Hz is truncated)

■ Suffix code
None: Hz(10^0)
HZ: Hz(10^0)
KHZ,KZ: kHz(10^3)
MHZ,MZ: MHz(10^6)
GHZ,GZ: GHz(10^9)

■ Initial setting 8.5KHZ: 8.5kHz

■ Example ADJINBW△8.5kHz

AMB

AMB A – B → A

■ Function Finds the difference between Trace-A and Trace B, and saves the result in Trace-B.

Header	Program command	Query	Response
AMB	AMB△sw	AMB?	sw sw=0,1

■ Value of sw 1,ON:On
Ø,OFF:Off
■ Suffix code None
■ Initial setting OFF
■ Example AMB△ON

AMBPL

AMBPL Normalize(A – B + DL → A)

■ Function Performs normalization (Trace-A – Trace-B + Display line level -> Trace-A).

Header	Program command	Query	Response
AMBPL	AMBPL△sw	AMBPL?	sw

■ Value of sw 1,ON:On
Ø,OFF:Off
■ Suffix code None
■ Initial setting OFF
■ Example AMBPL△ON

AMD

AMD Trace A Storage Mode

■ Function Selects the mode for processing the trace A waveform.

Header	Program command	Query	Response
AMD	AMD△n	AMD	AMD△n

■ Value of n
Ø: NORMAL
1: MAXHOLD
2: AVERAGE
3: MINHOLD
4: CUMULATIVE
5: OVERWRITE

■ Suffix code None

■ Initial setting Ø: NORMAL

■ Example AMD△Ø

APB

APB A + B → A

■ Function Adds Trace-A and Trace-B waveform data, and stores the result in Trace-B.

Header	Program command	Query	Response
APB	APB	———	———

■ Example APB

ARB

ARB Resolution Bandwidth

■ Function Switches the mode for setting the resolution bandwidth to AUTO or MANUAL

Header	Program command	Query	Response
ARB	ARB△sw	ARB?	ARB△sw

■ Value of sw
Ø: MANUAL
1: AUTO

■ Suffix code None

■ Initial setting 1: AUTO

■ Example
ARB△Ø
ARB△1

AST

AST Sweep Time

■ Function Switches the mode for setting the frequency sweep time to AUTO or MANUAL.

Header	Program command	Query	Response
AST	AST△sw	AST?	AST△sw

■ Value of sw
Ø: MANUAL
1: AUTO

■ Suffix code None

■ Initial setting 1: AUTO

■ Example
AST△Ø
AST△1

ASWT

ASWT Auto Sweep Time

■ Function Sets the AUTO SWEEP TIME

Header	Program command	Query	Response
ASWT	ASWT△sw sw=FAST, SLOW	ASWT?	sw sw=FAST, SLOW

■ Value of sw
FAST: FAST
SLOW: NORMAL

■ Suffix code None

■ Initial setting SLOW (provided te adress already allocated is not initialized)

■ Example
ASWT△FAST
ASWT△SLOW

AT

AT RF Attenuator

■ Function Sets the RF attenuator.

Header	Program command	Query	Response
AT	AT△a AT△n	AT?	n

■ Value of a
AUTO: AUTO
UP: UP
DN: DOWN

■ Value of n Ø to 7Ø (1Østep): 0 to 70dB(10dB step)

■ Suffix code
None: dB
DB : dB

■ Initial setting ATT=Calculated value when AUTO is selected for ATT

■ Example
AT△1Ø
AT△5Ø

ATB

ATB Trace-A → Trace-B

■ Function Copies the waveform data of Trace-A onto Trace-B.

Header	Program command	Query	Response
ATB	ATB	———	———

■ Example ATB

ATT

ATT RF Attenuator

■ Function Sets the RF attenuator.

Header	Program command	Query	Response
ATT	ATT△n	ATT?	ATT△n

■ Value of n

Ø:	0dB	12:	60dB
1:	10dB	13:	70dB
2:	20dB		
3:	30dB		
4:	40dB		
5:	50dB		

■ Suffix code None

■ Initial setting Calculated value when AUTO is selected for ATT

■ Example ATT△1

ATUN

ATUN Auto Tune

■ Function Detects the maximum peak point in the specified frequency band of the BG (background) band, and displays its spectrum in the center of the screen in CENTER-SPAN mode.

Header	Program command	Query	Response
ATUN	ATUN	————	————

■ Example `ATUN`

AUNITS

AUNITS Unit for Log Scale

■ Function Sets the display units when the LOG scale is selected.

Header	Program command	Query	Response
AUNITS	AUNITS△a	AUNITS?	a

■ Value of a
`DBM :` dBm
`DBUV:` dBμV
`DBMV:` dBmV
`DBUVE:` dBmV(emf)
`V:` V
`W:` W

■ Suffix code None

■ Initial setting `DBM:` dBm (provided the address already allocated is not initialized)

■ Example
```
AUNITS△DBM
AUNITS△V
```

AUTO

AUTO Coupled Function All Auto

■ Function Executes all coupled functions (RBW, VBW, SWT, and ATT) in AUTO mode.

Header	Program command	Query	Response
AUTO	AUTO	————	————

■ Example AUTO

AVB

AVB Video Bandwidth

■ Function Switches the mode for setting the video bandwidth to AUTO or MANUAL.

Header	Program command	Query	Response
AVB	AVB△n	AVB?	AVB△n

■ Value of n
Ø: MANUAL
1: AUTO
2: OFF

■ Suffix code None

■ Initial setting 1: AUTO

■ Example
AVB△Ø
AVB△1

AVGPAUSE

AVGPAUSE Average Sweep Mode

■ Function
Specifies the processing (pause or continue) executed after the specified average sweeps.

Header	Program command	Query	Response
AVGPAUSE	AVGPAUSE△sw	AVGPAUSE?	sw sw=0,1

■ Value of sw
Ø,OFF: Continue
1,ON: Pause

■ Suffix code
None

■ Initial setting
ON: Pause

■ Example
AVGPAUSE△ON

AVR

AVR Number of Trace Average

■ Function
Sets the averaging rate (number of sweep repetitions).

Header	Program command	Query	Response
AVR	AVR△n	AVR?	AVR△n

■ Value of n
Ø: 4times
1: 8times
2: 16times
3: 32times
4: 128times

■ Suffix code
None

■ Initial setting
1: 8times

■ Example
AVR△Ø
AVR△3

AWR

AWR Trace A Write Switch

■ Function Controls writing of the waveform data to trace A.

Header	Program command	Query	Response
AWR	AWR△sw SW=ON,1,OFF,0	AWR?	AWR△sw sw=ON,OFF

■ Value of sw 1,ON: TRACE A WRITE ON (same function as CLRW△TRA)
Ø,OFF: TRACE A WRITE OFF(same function as VIEW△TRA)
■ Suffix code None
■ Initial setting 1: TRACE A WRITE ON
■ Example AWR△0

AXB

AXB Exchange Trace-A and Trace-B

■ Function Exchanges the waveform data of Trace-A and Trace-B.

Header	Program command	Query	Response
AXB	AXB	————	————

■ Example AXB

B1

B1 Trace B Write ON

■ Function Clears the trace B waveform data to set the write mode to ON (same function as BWR△1, CLRW△TRB).

Header	Program command	Query	Response
B1	B1	———	———

■ Example B1

B2

B2 Trace B Max Hold

■ Function Allows the trace B waveform to be processed in MAX HOLD mode (same function as BMD△1).

Header	Program command	Query	Response
B2	B2	———	———

■ Example B2

BAUD

BAUD Baud rate

■ Function Changes the baud rate of the RS232C.

Header	Program command	Query	Response
BAUD	BAUD△n	BAUD?	n

■ Value of n
1200:1200 BPS
2400:2400 BPS
4800:4800 BPS
9600:9600 BPS

■ Suffix code None

■ Initial setting 2400:2400 BPS

■ Example BAUD△9600

BGWR

BGWR Trace BG Write Switch

■ Function Controls writing of the waveform data to trace BG.

Header	Program command	Query	Response
BGWR	BGWR△sw	BGWR?	BGWR△sw sw=ON,OFF

■ Value of sw
1,ON: TRACE BG WRITE ON (same function as CLRW△TRBG)
0,OFF: TRACE BG WRITE OFF (same function as VIEW△TRBG)

■ Suffix code None

■ Initial setting ON: TRACE BG WRITE ON

■ Example BGWR△ON

BIN

BIN ASCII / Binary Data Out

■ Function Sets the format of output trace data to ASCII or BINARY.

Header	Program command	Query	Response
BIN	BIN△sw	————	————

■ Value of sw
Ø,OFF: ASCII
1,ON: BINARY

■ Suffix code None

■ Initial setting Ø: ASCII

■ Example
BIN△Ø
BIN△ON

■ Restriction according to model type and option
When RS-232C interface is used, it is not enabled to use the trace data output of BINARY format.

BMD

BMD Trace B Storage Mode

■ Function Selects the mode for processing the trace B waveform.

Header	Program command	Query	Response
BMD	BMD△n	BMD?	BMD△n

■ Value of n
Ø: NORMAL
1: MAX HOLD
2: AVERAGE
3: MIN HOLD
4: CUMULATIVE
5: OVER WRITE

■ Suffix code None

■ Initial setting Ø: NORMAL

■ Example BMD△Ø

BND

BND Band Select

■ Function Sets the band.

Header	Program command	Query	Response
BND	BND△n	BND?	BND△n

■ Value of n

	(MS2665C)	(MS2667C)	(MS2668C)
Ø: BAND AUTO=	0 to 21.2 GHz	0 to 30.0 GHz	0 to 40.0 GHz
1: BAND 0=	0 to 3.2 GHz	0 o 3.2 GHz	0 o 3.2 GHz
2: BAND 1^{-}=	2.92 to 6.5 GHz	3.1 to 6.5 GHz	3.1 to 5.6 GHz
3: BAND 1^{+}=	6.4 to 8.1 GHz	6.4 to 8.1 GHz	5.4 to 8.1 GHz
4: BAND 2^{+}=	8.0 to 15.3 GHz	8.0 to 15.3 GHz	7.9 to 14.3 GHz
5: BAND 3^{+}=	15.2 to 21.2 GHz	15.2 to 22.4 GHz	14.1 to 26.5 GHz
6: BAND 4^{+}=	-----	22.3 to 30.0 GHz	26.2 to 40.0 GHz

■ Suffix code None

■ Initial setting AUTO: BAND AUTO= 0 Hz to 21.2 GHz/30.0 GHz/40 GHz

■ Example BND△Ø
BND△3

■ Restrictions according to model type and options

If equipment is MS2665C, n=6 can not be selected.

BNDC

BNDC Band Select

■ Function Sets the band.

Header	Program command	Query	Response
BNDC	BNDC△a a=AUTO,0,1^-,1^+,2^+,3^+,4^+, 1^{++},2^-,3^-	BNDC?	a a=AUTO,0,1^-,1^+,2^+,3^+,4^+, 1^{++},2^-,3^-

■ Value of a

		(MS2665C)	(MS2667C)
AUTO:	BAND AUTO=	0 to 8.1 GHz	0 to 30.0 GHz
Ø:	BAND 0=	0 to 3.2 GHz	0 to 3.2 GHz
1-:	BAND 1^-=	2.92 to 6.5 GHz	3.1 to 6.5 GHz
1+:	BAND 1^+=	6.4 to 8.1 GHz	6.4 to 8.1 GHz
2+:	BAND 2^+=	8.0 to 15.3 GHz	8.0 to 15.3 GHz
3+:	BAND 3^+=	15.2 to 21.2 GHz	15.2 to 22.4 GHz
4+:	BAND 4^+=	-----	22.3 to 30.0 GHz

		(MS2668C)
AUTO:	BAND AUTO=	0 to 40.0 GHz
Ø:	BAND 0=	0 to 3.2 GHz
1-:	BAND 1^-=	3.1 to 5.6 GHz
1+:	BAND 1^+=	5.4 to 8.1 GHz
1++:	BAND 1^{++}=	7.9 to 14.3 GHz
2–:	BAND 2^-=	14.1 to 26.5 GHz
3–:	BAND 3^-=	26.2 to 40.0 GHz

■ Suffix code None

■ Initial setting AUTO: BAND AUTO= 0 Hz to 21.2 GHz/30.0 GHz/40.0GHz

■ Example

```
BNDC△AUTO
BNDC△1+
```

■ Restrictions according to model type and options

If equipment is MS2665C, a=0, 1–, 2+, 3+ can be selected.
If equipment is MS2667C, a=0, 1–, 1+, 2+, 3+, 4+ can be selected.
If equipment is MS2668C, a=0, 1–, 1+, 1++, 2–, 3– can be selected.

BRIGHT

BRIGHT Adjust Brightness

■ Function Selects the LCD display brightness.

Header	Program command	Query	Response
BRIGHT	BRIGHT△n	BRIGHT?	n

■ Value of n 1 to 4
■ Suffix code None
■ Example BRIGHT△3

BSAUTO

BSAUTO BW / SWT Auto

■ Function Allows RBW, VBW, and the sweep time to be set in AUTO mode.

Header	Program command	Query	Response
BSAUTO	BSAUTO	———	———

■ Example BSAUTO

BTA

BTA Trace-B→Trace-A

■ Function Copies the data of the Trace-B waveform to Trace-A.

Header	Program command	Query	Response
BTA	BTA	———	———

■ Example BTA

BWR

BWR Trace B Write Switch

■ Function Controls writing of the waveform data to trace B.

Header	Program command	Query	Response
BWR	BWR△sw	BWR?	BWR△sw sw=ON,OFF

■ Value of sw
1,ON: TRACE B WRITE ON (same function as CLRW△TRB)
Ø,OFF: TRACE B WRITE OFF (same function as VIEW△TRG)

■ Suffix code None

■ Initial setting 1: TRACE B WRITE ON

■ Example BWR△Ø

C1

C1 A - B Off

■ Function Turns the A-B function to OFF.

Header	Program command	Query	Response
C1	C1	———	———

■ Example C1

C2

C2 A - B On

■ Function Turns the A-B function to ON.

Header	Program command	Query	Response
C2	C2	———	———

■ Example C2

CA

CA RF Attenuator Auto

■ Function Sets the attenuator to AUTO mode (same function as AAT1, AT△AUTO).

Header	Program command	Query	Response
CA	CA	————	————

■ Example CA

CAL

CAL Calibration

■ Function Performs calibration using the internal CAL signal.

Header	Program command	Query	Response
CAL	CAL△n	————	————

■ Value of n
Ø: All
1: Frequency
2: Level
3: FM

■ Suffix code None

■ Example CAL△Ø

CDT

CDT Set Correction factor on

■ Function Controls correction of the frequency characteristics.

Header	Program command	Query	Response
CDT	CDT△sw	CDT?	CDT△sw SW=0,1

■ Value of sw Ø,OFF: Off
1,ON: On
■ Suffix code None
■ Initial setting Ø: Off
■ Example CDT△1

CF

CF Center Frequency

■ Function Sets the center frequency (same function as CNF).

Header	Program command	Query	Response
CF	CF△f CF△a	CF?	f f=-100000000 to 40000000000 Transfers the data with no suffix code in units of 1 Hz.

■ Value of f –100MHz to 40.0GHz
■ Value of a UP: CENTER FREQSTEP UP (same function as FUP)
DN: CENTER FREQSTEP DOWN (same function as FDN)
■ Suffix code f: None: Hz(10^0)
HZ: HZ(10^0)
KHZ,KZ kHz(10^3)
MHZ,MZ MHz(10^6)
GHZ,GZ GHz(10^9)
a: None
■ Initial setting Initial value of f = 10.6 GHz (MS2665C), 15.0 GHz (MS2667C), 20.0 GHz (MS2668C)
■ Example CF△1235456
CF△5ØMHz
CF△UP
■ Restrictions according to model type and options

If equipment is MS2665C, upper limit of f is equal to 21.2 GHz

If equipment is MS2667C, upper limit of f is equal to 30.0 GHz

CHPWRFACT

CHPWRFACT   Channel Power Correction Factor

■ Function   Sets the Channel power correction factor.

Header	Program command	Query	Response
CHPWRFACT	CHPWRFACT△l	CHPWRFACT?	l

■ Value of l   –99.99dB to 99.99dB
■ Suffix code   None: dB
DB, DBM, DM: dB
■ Initial setting   Ø: 0dB
■ Example   CHPWRFACT△-2.5DB

CLRMENU

CLRMENU   Clear menu define

■ Function   Initializes the data defined on the menu.

Header	Program command	Query	Response
CLRMENU	CLRMENU	———	———

■ Example   CLRMENU

CLRW

CLRW Clear & Write

■ Function Clears the trace waveform data to set the write mode to ON.

Header	Program command	Query	Response
CLRW	CLRW△tr	————	————

■ Value of tr
`TRA:` Trace A (same function as AWR△1)
`TRB:` Trace B (same function as BWR△1)
`TRBG:` Trace BG (same function as BGWR△1)
`TRTIME:` Trace TIME (same function as TMWR△1)

■ Example `CLRW△TRA`

CMK?

CMK? Current Marker Position

■ Function Reads the current marker position.

Header	Program command	Query	Response
CMK?	————	CMK?	CMK△p

■ Value of p 0 to 500

■ Example `CMK?`

CNF

CNF Center Freqency

■ Function Sets the center frequency (same function as CF).

Header	Program command	Query	Response
CNF	CNF△f	CNF?	CNF△f f=–100000000 to 40000000000 Transfers the data with no suffix code in units of 1 Hz.

■ Value of f –100MHz to 40.0 GHz
■ Suffix code
None: Hz(10^0)
HZ: HZ(10^0)
KHZ,KZ: kHz(10^3)
MHZ,MZ: MHz(10^6)
GHZ,GZ: GHz(10^9)
■ Initial setting Value of f = 10.6 GHz (MS2665C), 15.0 GHz (MS2667C), 20.0 GHz (MS2668C)
■ Example
CNF△123456
CNF△50MHZ
CNF?
■ Restrictions according to model type and options
If equipment is MS2665C, upper limit of f is equal to 21.2 GHz
If equipment is MS2667C, upper limit of f is equal to 30.0 GHz

CNVLOSS

CNVLOSS EXT Mixer Loss

■ Function Sets a conversion loss value of the external mixer; a range from 0.00 to 99.99 dB can be set with an increment of 0.01 dB.
(Setting can be performed on the measurement band currently selected.)

Header	Program command	Query	Response
CNVLOSS	CNVLOSS△c c=0.00 to 99.99	CNVLOSS?	C c=0.00 to 99.99 dB

■ Value of c 0.00 to 99.99 dB (0.01 dB step)
■ Suffix code
None: dB
DB: dB
■ Initial setting Initial value of c=18.00 dB
■ Example CNVLOSS△20
■ Restrictions according to model type and options
This command is a MS2667C/68C dedicated command.

COLORDEF

COLORDEF Define user color pattern

■ Function Sets each frame color of user definition patterns.

Header	Program command	Query	Response
COLORDEF	COLORDEF△n,r,g,b	COLORDEF?△n	r,g,b

■ Value of n 0 to 16: Frame number
■ Value of r,g,b 0 to 63: Strength of the display color of r(red), g(green), and b(blue)
■ Suffix code None
■ Initial setting Set value of color pattern 1
■ Example `COLORDEF△1,48,50,63`

COLORPTN

COLORPTN Color pattern

■ Function Selects the display color from the display color patterns.

Header	Program command	Query	Response
COLORPTN	COLORPTN△a	COLORPTN?	a

■ Value of a
`COLOR1:` Color pattern-1
`COLOR2:` Color pattern-2
`COLOR3:` Color pattern-3
`COLOR4:` Color pattern-4
`USERCOLOR:` User definition pattern
■ Suffix code None
■ Initial setting `COLOR1:` Color pattern-1
■ Example `COLORPTN△USERCOLOR`

COMMENT

COMMENT Comment display

■ Function Sets the display method for the comment column.

Header	Program command	Query	Response
COMMENT	COMMENT△a	COMMENT?	a

■ Value of a
TITLE: Displays the title.
TIME: Displays the time.
OFF: No comment is displayed.

■ Suffix code None

■ Initial setting OFF: No comment is displayed.

■ Example COMMENT△TITLE

COMP

COMP Composite Mode

■ Function Switching of the video signal from the Composite Out terminal at the rear panel is carried out by the following key operations.

Header	Program command	Query	Response
COMP	COMP△a	COMP?	a

■ Value of a
NRM: Normal
PAL: PAL
NTSC: NTSC

■ Suffix code None

■ Example COMP△PAL

CONTS

CONTS Continuous Sweep Mode

■ Function Sets the sweep mode to continuous mode (same function as S1).

Header	Program command	Query	Response
CONTS	CONTS		

■ Example `CONTS`

COPYCOLOR

COPYCOLOR Copy into user pattern from Color pattern

■ Function Selects the display color pattern, and copies it to the user definition pattern.

Header	Program command	Query	Response
COPYCOLOR	COPYCOLOR△a	————	————

■ Value of a

```
COLOR1:  Color pattern-1
COLOR2:  Color pattern-2
COLOR3:  Color pattern-3
COLOR4:  Color pattern-4
```

■ Suffix code None

■ Example `COPYCOLOR△COLOR4`

CORC

CORC Correction Factor Initialization

■ Function Initializes the correction factor currently selected by the CORR command.

Header	Program command	Query	Response
CORC	CORC	————	————

■ Example CORC
All frequency data and level data are initialized. The initialized data is used as the 0 dB correction values in each frequency range.

CORD

CORD Correction Factor Entry

■ Function Registers the correction factor currently selected by the CORR command.
If the correction factor is set to OFF, it is not valid.

Header	Program command	Query	Response
CORD	CORD△n,f,l n=0 to 149 f=0 to 400GHz l=–100.00 to +100.00dB (incremented in 0.01 dB steps)	CORD?△n	CORD△f,l f = 0 to 400 000 000 000 (no units) l= –100.00 to +100.00 dB (incremented in 0.01 steps)

■ Value of n 0 to 149
■ Value of f 0 to 400GHz
■ Value of ℓ –100.00 to +100.00 dB (incremented in 0.01 dB steps)
■ Suffix code

f:	None:	Hz(10^0)
	HZ:	HZ(10^0)
	KHZ,KZ:	kHz(10^3)
	MHZ,MZ:	MHz(10^6)
	GHZ,GZ:	GHz(10^9)
l:	None:	dB
	DB:	dB

■ Example CORD△Ø,1MHZ,1Ø
CORD△1,2ØØØØØ,1Ø
If fn –1<fn<fn +1 is not satisfied when n–1<n<n+1, an error occurs.

CORR

CORR Correction Factor Select

■ Function Selects the type of correction factor.

Header	Program command	Query	Response
CORR	CORR△n	CORR?	CORR△n

■ Value of n
Ø,OFF: OFF
1: CORR1
2: CORR2
3: CORR3
4: CORR4
5: CORR5

■ Suffix code None

■ Initial setting Ø: OFF (the correction factor already registered is not initialized)

■ Example
CORR△Ø
CORR△2
CORR△4

CORRLABEL

CORRLABEL Correction Factor Label

■ Function Registers the name of the correction factor currently selected by the CORR command.

Header	Program command	Query	Response
CORRLABE	CORRLABEL△n,text	CORRLABEL?△n	"text"

■ Value of n 1 to 5

■ Value of text String of up to 30 characters enclosed by single or double quotes.

■ Suffix code None

■ Example
CORRLABEL△1,"CORRECTION FACTOR"
CORRLABEL△2,'MS2665C'

CORRLOAD

CORRLOAD Load Correction data

■ Function Reads the correction data from the memory card file.

Header	Program command	Query	Response
CORRLOAD	CORRLOAD△n	————	————

■ Value of n 1 to 99
■ Suffix code None
■ Example CORRLOAD△1

CORRSAVE

CORRSAVE Save Correction data

■ Function Saves the internal correction data to the memory card.

Header	Program command	Query	Response
CORRSAVE	CORRSAVE△n	————	————

■ Value of n 1 to 99
■ Suffix code None
■ Example CORRSAVE△1

COUPLE

COUPLE　　Coupling Mode

■ Function　Switches the coupling to AC or DC to monitor an FM waveform.

Header	Program command	Query	Response
COUPLE	COUPLE△a	COUPLE?	a

■ Value of a　AC:　AC COUPLING
DC:　DC COUPLING

■ Suffix code　None

■ Initial setting　AC:　AC COUPLING

■ Example　COUPLE△AC
COUPLE△DC

CR

CR　　Resolution Bandwidth Auto

■ Function　Sets the resolution bandwidth selection to the AUTO mode
(same function as ARBV△1, RB△AUTO).

Header	Program command	Query	Response
CR	CR	———	———

■ Example　CR

CRS

CRS Count Resolution

■ Function Selects the resolution of the frequency counter.

Header	Program command	Query	Response
CRS	CRS△n	CRS?	CRS△n

■ Value of n
Ø: 1Hz
1: 10Hz
2: 100Hz
3: 1kHz

■ Suffix code None

■ Initial setting 3: 1kHz

■ Example
CRS△0
CRS△3

CT

CT Sweep Time Auto

■ Function Sets the frequency sweep time to AUTO mode (same function as AST△1, ST△AUTO).

Header	Program command	Query	Response
CT	CT	———	———

■ Example CT

CV

CV Video Bandwidth Auto

■ Function Sets the video bandwidth to AUTO mode
(same function as AVB△1, VB△AUTO).

Header	Program command	Query	Response
CV	CV	———	———

■ Example CV

DATB

DATB Data bit

■ Function Specifies the data length of the RS232C.

Header	Program command	Query	Response
DATB	DATB△n	DATB?	n

■ Value of n 7: 7 bit
8: 8 bit
■ Suffix code None
■ Initial setting 8: 8 bit
■ Example DATB△7

DATE

DATE Date

■ Function Sets the built-in clock of the spectrum analyzer to the specified date.

Header	Program command	Query	Response
DATE	DATE△yyyy,mm,dd	DATE?	yyyy,mm,dd

■ Value of yyyy 1960 to 2059 (year)
■ Value of mm 01 to 12 (month)
■ Value of dd 01 to 31 (day)
■ Suffix code None
■ Example DATE△1997,Ø3,31

DATEMODE

DATEMODE Date Display mode

■ Function Sets the display method for the date display column.

Header	Program command	Query	Response
DATEMODE	DATEMODE△a	DATEMODE?	a

■ Value of a
YMD: Year/month/date
DMY: Day-month-year
MDY: Month-day-year

■ Suffix code None

■ Initial setting YMD: Year/month/day

■ Example DATEMODE△MDY

DET

DET Detection Mode

■ Function Selects the detection mode for the waveform data being displayed.

Header	Program command	Query	Response
DET	DET△d	DET?	d d=POS,SMP,NEG

■ Value of d
Ø: POSITIVE PEAK
1: SAMPLE
2: NEGATIVE PEAK
3: NORMAL
POS: POSITIVE PEAK
SMP: SAMPLE
NEG: NEGATIVE PEAK
NRM: NORMAL

■ Suffix code None

■ Initial setting Ø: POSITIVE PEAK

■ Example
DET△0
DET△SMP

DETM

DETM Detection Mode

■ Function Selects the detection mode for the specified trace.

Header	Program command	Query	Response
DETM	DETM△tr,a	DETM?△tr	a

■ Value of tr
- TRA: Trace A
- TRB: Trace B
- TRIME: Trace TIME

■ Value of a
- POS: POSITIVE PEAK
- SMP: SAMPLE
- NEG: NEGATIVE PEAK
- NRM: NORMAL

■ Suffix code None

■ Initial setting POS: POSITIVE PEAK

■ Example
```
DETM△TRA,POS
DETM△TRB,SMP
DETM△TRIME,SMP
```

DFMT

DFMT Display Format

■ Function Specifies the display mode/format.

Header	Program command	Query	Response
DFMT	DFMT△a	DFMT?	a

■ Value of a
A: Trace A
B: Trace B
TIME: Trace TIME
AB1: Trace A/Trace B (A & B)
AB2: Trace A/Trace B (A/B)
AB3: Trace A/Trace B (A<B)
ABG1: Trace A/Trace BG (BG>A)
ABG2: Trace A/Trace BG (BG<A)
ATIME1: Trace A/Trace TIME (TIME>A)
ATIME2: Trace A/Trace TIME (TIME<A)

■ Suffix code None
■ Initial setting A: Trace A
■ Example DFMT△TIME

DIM

DIM Dimensional common variable

■ Function Declares array common variable for PTA.

Header	Program command	Query	Response
DIM	DIM△a,n[,m]	———	———

■ Value of a Array common variable name(integer/real-number numerical variable name, alpha-numerical characters of less than 7 characters)
■ Value of n 1 to 1024: One-dimensional array size
■ Value of m 1 to 1024: Two-dimensional array size, omittable
■ suffix code None
■ Example DIM△ABC,10,0 --- Declares DIM @ABC(10).
DIM△DEF%,20 --- Declares DIM @DEF%(20).
DIM△GHI,5,5 --- Declares DIM @GHI(5,5).

DISPLAY

DISPLAY LCD Display On/Off

■ Function Specifies whether the LCD display is on or off.

Header	Program command	Query	Response
DISPLAY	DISPLAY△sw	————	————

■ Value of sw OFF: LCD display is off.
ON: LCD display is on.
■ Suffix code None
■ Initial setting ON: LCD display is on.
■ Example DISPLAY△OFF

DL

DL Display line,Display-line Level

■ Function Turns the display line on or off, and sets its level.

Header	Program command	Query	Response
DL	DL△sw DL△l	DL?	OFF l: A vailable for the current scale unit, provided that μV units are selected for V, and W units are selected for W.

■ Value of sw
ON: ON
OFF: OFF

■ Value of ℓ
Value equivalent to full scale of current Y-axis.
For LOG scale: RLV-100 to RLV
For LIN scale: 0 to RLV.
For A-B: –100.00 to 100.00 dB
For FM monitor at Trace-time: –Max range to +Max range

■ Suffix code
None: Available for the current scale unit, provided V units are always selected in LIN mode.
DB, DBM, DM: dBm
DBMV: dBmV
DBUV: dBμV
DBUVE: dBμV (emf)
V: V
MV: mV
UV: uV
W: W
MW: mW
UW: μW
NW: nW
PW: pW
FW: fW

■ Initial setting –60.00 dBm(Level equivalent to center point of the scale)

■ Example
DL△OFF
DL△-1Ø.ØDBM

DLT

DLT Time Delay

■ Function Sets the delay time.

Header	Program command	Query	Response
DLT	DLT△t	DLT?	DLT△t

■ Value of t –1000s to 65.5ms
■ Suffix code US: µs
MS: ms
S: s
■ Initial setting Ø: s
■ Example DLT△-20MS

DOWNLOAD

DOWNLOAD Download PTA-library name

■ Function Starts the registration of the PTA library.

Header	Program command	Query	Response
DOWNLOAD	DOWNLOAD△a	———	———

■ Value of a PTA-library name of less than 8 characters
■ Suffix code None
■ Example DOWNLOAD△SAMPLE1

DSPLV

DSPLV Marker Level Absolute ; Relative

■ Function — Specifies the marker level in the absolute value display or in the relative value display when seen from the display line.

Header	Program command	Query	Response
DSPLV	DSPLV△a	DSPLV?	a

■ Value of a
ABS: Absolute value
REL: Relative value

■ Suffix code — None

■ Initial setting — ABS: Absolute value

■ Example — DSPLV△REL

DSPLVM

DSPLVM Marker Level Absolute/Relative

■ Function — With the trace mode specified, also specifies the marker level in the absolute value display or in the relative value display when seen from the display line.

Header	Program command	Query	Response
DSPLVM	DSPLVM△tr,a	DSPLVM?△tr	a

■ Value of tr
TRA: Trace A
TRB: Trace B
TRIME: Trace Time
TRBG: Trace BG

■ Value of a
ABS: Absolute value
REL: Relative value

■ Suffix code — None

■ Initial setting — ABS: Absolute value

■ Example — DSPLVM△TRA,REL

DVAR

DVAR Write value to dimensional common variable

■ Function: Write a value at array common variable for PTA.

Header	Program command	Query	Response
DVAR	DVAR△a,n,m,d	DVAR?△a,n,m	d

■ Value of a: Array common variable name(integer/real-number numerical variable name, alpha-numerical characters of less than 7 characters)

■ Value of n: 1 to 1024: One-dimensional array size

■ Value of m: –1, 1 to 1024: Two-dimensional array size, omittable

■ Value of d: Value to be substituted (integer or real-number)

■ Example:

DVAR△ABC,5,–1,1.2345 --- @ABC(5)=1.2345
DVAR△DEF%,15,–1,200 --- @DEF%(15)=200
DVAR△GHI,2,3,–54.3 --- @GHI(2,3)=–54.3

E1

E1 Peak Search

■ Function Executes the function for peak search (same function as MKS△0,MKMP).

Header	Program command	Query	Response
E1	E1	————	————

■ Example E1

E2

E2 Marker to CF

■ Function Sets the marker to the center frequency (same function as MKR△3, MKCF).

Header	Program command	Query	Response
E2	E2	————	————

■ Example E2

E3 Marker to CF Step Size

■ Function Sets the marker to the frequency step size (same function as MKR△5M, MKSS).

Header	Program command	Query	Response
E3	E3	————	————

■ Example E3

E4 Marker to REF

■ Function Sets the marker to the reference level (same function as MKR△4, MKRL).

Header	Program command	Query	Response
E4	E4	————	————

■ Example E4

ECYC

ECYC Event Cyclical

■ Function Sets the generation period of event interruption for PTA.

Header	Program command	Query	Response
ECYC	ECYC△t	————	————

■ Value of t 0 to 3600 (s, 0.1 s resolution)
For 0, event is not generated.
■ Suffix code None
■ Example ECYC△2

EDLY

EDLY Event Cyclical

■ Function Event Delay for PTA.

Header	Program command	Query	Response
EDLY	EDLY△t	————	————

■ Value of t 0 to 3600 (s, 0.1 s resolution)
For 0, event is not generated.
■ Suffix code None
■ Example EDLY△3Ø

ENTRY

ENTRY Open entry

■ Function Specifies the entry (prompt for input).

Header	Program command	Query	Response
ENTRY	ENTRY△text,n,a	ENTRY?	b

■ Value of text Input prompt: String of up to 20 characters enclosed by single or double quotes.
HZ-system numeric key + data knob + Step key

■ Value of n Ø,1 to 16: Input type
0: Deletion of input prompt
1: Hz-system numeric key + data knob + Step key
2: Hz-system numeric key + data knob
3: Hz-system numeric key + Step key
4: Hz-system numeric key
5: sec/V/W-system numeric key + data knob + Step key
6: sec/V/W-system numeric key + data knob
7: sec/V/W-system numeric key + Step key
8: sec/V/W-system numeric key
9: dB-system numeric key + data knob + Step key
10: dB-system numeric key + data knob
11: dB-system numeric key + Step key
12: dB-system numeric key
13: No-unit-system numeric key + data knob + Step key
14: No-unit-system numeric key + data knob
15: No-unit-system numeric key + Step key
16: No-unit-system numeric key
Hz-system numeric key: Valid for Hz/kHz/MHz/GHz keys
sec/V/W-system numeric key: Valid for μs/ms/s, μV/mV/V, and μW/mW/W keys
dB-system numeric key: Valid for Enter/dB keys
No-unit-system numeric key: Valid for Enter key

■ Value of a Display of current value

■ Value of b Input data (value of each input type)
Numeric input: Converted numeric data according to unit key
Input type 1 to 4: 1 Hz unit
5 to 8: 1 ns / 1 nV / 1 nW unit
9 to 12: 0.01 dBm / 0.01 dB Unit
13 to 16: input data as it is
Step up key: "STEP△UP"
Step down key: "STEP△DOWN"
Data knob counterclockwise: "KNOB△LEFT"
Data knob clockwise: "KNOB△RIGHT"
Input cancelled: "***"
Double entry opened: "%%%"

■ Suffix code None

■ Example ENTRY△"enclosed Channel=",13,"1"
ENTRY?

ERASEWUP

ERASEWUP Erase warm up message

■ Function Erases the message of warm up.

Header	Program command	Query	Response
ERASEWUP	ERASEWUP	———	———

■ Example ERASEWUP

ERROR?

ERROR? Read out error code

■ Function Reads the contents of error codes, for example, details of an execution error.

Header	Program command	Query	Response
ERROR?	———	ERROR?	e1,e2

■ Value of e1,e2 Main code and subcode which indicate the error details.

Main code
300 to 399: Syntax error
400 to 499: Communication error
450 to 459: Media error
500: Range error
501: Inhibit error
502: Execution error
503: Setting condition not enough
504: Hardware error
600: Warning

ESE2

ESE2 Event Status Enable(END)

■ Function
Allows the END Event Status Enable Register to select which bit in the corresponding Event Register causes a TRUE ESB summary message bit 2 when set.

Header	Program command	Query	Response
ESE2	ESE2△n	ESE2?	n

■ Value of n
Ø to 255: Represents the sum of the bit-weighted values enabled by the $2^0=1, 2^1=2, 2^2=4, 2^3=8, 2^4=16, 2^5=32, 2^6=64, 2^7=128$ corresponding to bits 0, 1, 2, 3, 4, 5, 6, 7 of theEND Event Status Register.

■ Suffix code
None

■ Example
ESE2△1

ESR2?

ESR2? Event Status Regiser(END)

■ Function
Allows the sum of the binary-weighted event bit values of the END Event Status Register to be read out by converting them to decimal. After readout, the END Event Status Register is reset to 0.

Header	Program command	Query	Response
ESR2?	————	ESR2?	n

■ Value of n
0 to 255

■ Suffix code
None

■ Example
ESR2?

ETIM

ETIM Event Time

■ Function Sets the time of event-interruption generation for PTA.

Header	Program command	Query	Response
ETIM	ETIM△t1,t2,t3	————	————

■ Value of t1 to t3

t1: Hour (0 to 23)
t2: Minute(0 to 59)
t3: Second(0 to 59)

■ Suffix code None

■ Example ETIM△1Ø,15,3Ø

EX

EX Exchange Trace-A and Trace-B

■ Function Exchanges the trace-A and trace-B wave data.

Header	Program command	Query	Response
EX	EX	————	————

■ Example EX

EXTTYPE

EXTTYPE Ext Trigger Input Type

■ Function Chooses the level of the external trigger when EXT is selected for the trigger source.

Header	Program command	Query	Response
EXTTYPE	EXTTYPE△a	EXTTYPE?	a

■ Value of a 1ØV: ±10V input Level
TTL: TTL input Level
■ Suffix code None
■ Initial setting 1ØV: ±10V input Level
■ Example EXTTYPE△10V
EXTTYPE△TTL

FA

FA Start Frequency

■ Function Sets the start frequency (same function as STF).

Header	Program command	Query	Response
FA	FA△f	FA?	f f=–100000000 to 0 to 40000000000 Transfers the data with no suffix code in units of 1 Hz.

■ Value of f –100MHz to 40.0GHz
■ Suffix code
None: Hz(10^0)
HZ: Hz(10^0)
KHZ,KZ: kHz(10^3)
MHZ,MZ: MHz(10^6)
GHZ,GZ: GHz(10^9)
■ Initial setting Initial value of f = 0 Hz
■ Example FA△1GZ
■ Restrictions according to model type and options
If equipment is MS2665C, upper limit of f is equal to 21.2 GHz.
If equipment is MS2667C, upper limit of f is equal to 30.0 GHz.

FB

FB Stop Frequency

■ Function Sets the stop frequency (same function as SOF).

Header	Program command	Query	Response
FB	FB△f	FB?	f f=–100000000 to 0 to 40000000000 Transfers the data with no suffix code in units of 1 Hz.

■ Value of f –100MHz to 40.0GHz
■ Suffix code
None: Hz(10^0)
HZ: Hz(10^0)
KHZ,KZ: kHz(10^3)
MHZ,MZ: MHz(10^6)
GHZ,GZ: GHz(10^9)
■ Initial setting Initial value of f = 21.2 GHz (MS2665C), 30.0 GHz (MS2667C), 40.0 GHz (MS2668C)
■ Example FB△2GHZ
■ Restrictions according to model type and options
If equipment is MS2665C, upper limit of f is equal to 21.2 GHz.
If equipment is MS2667C, upper limit of f is equal to 30.0 GHz.

FCAL10

FCAL10 Frequency Cal On/Off

■ Function Specifies whether the Freq Cal is performed.

Header	Program command	Query	Response
FCAL10	FCAL10△sw	FCAL10?	sw

■ Value of sw 1: On
Ø: Off
■ Suffix code None
■ Initial setting 1: On
■ Example FCAL1Ø△Ø

FDN

FDN Center Frequency Step Down

■ Function Decreases the center frequency by the frequency step size if it has been set (same function as CF△DN).

Header	Program command	Query	Response
FDN	FDN	—	—

■ Example FDN

FMRNG

FMRNG FM Range

■ Function Sets the bandwidth for demodulating FM when trace TIME is selected for FM monitoring.

Header	Program command	Query	Response
FMRNG	FMRNG△f	FMRNG?	f f=2000 to 200000 Transfers the data with no suffix code in units of 1 Hz.

■ Value of f 2kHz to 200kHz : 2kHz/div to 200kHz/div

■ Suffix code
None: Hz/div
HZ: Hz/div
KHZ,KZ: kHz/div
MHZ,MZ: MHz/div
GHZ,GZ: GHz/div

■ Initial setting 2ØØkHz/div

■ Example FMRNG△2ØKHZ

FOFFSET

FOFFSET Frequency Offset

■ Function Sets the frequency offset value.

Header	Program command	Query	Response
FOFFSET	FOFFSET△c	FOFFSET?	C=0 to 100GHz

■ Value of c 0Hz to 100GHz (1MHz step)

■ Suffix code
None: Hz (10∧0)
HZ: Hz (10∧0)
KHZ: kHz (10∧3)
MHZ: MHz (10∧6)
GHZ: GHz (10∧9)

■ Initial setting ØHz

■ Example
FOFFSET△5ØØMHZ
FOFFSET?

■ Restrictions according to model type and options

This command is a MS2667C/68C dedicated command.

FOFMD

FOFMD Frequency Offset Mode

■ Function Turns the frequency offset ON/OFF.

Header	Program command	Query	Response
FOFMD	FOFMD△a	FOFMD?	a=0,1

■ Value of n 0,OFF: OFF
1,ON: ON
■ Suffix code None
■ Initial setting 0: OFF
■ Example FOFMD△Ø
FOFMD?

■ Restrictions according to model type and options

This command is a MS2667C/68C dedicated command.

FRQ

FRQ Frequency Mode

■ Function Selects the mode for setting the FG frequency band.

Header	Program command	Query	Response
FRQ	FRQ△n	FRQ?	FRQ△n

■ Value of n 0: CENTER-SPAN
2: START-STOP
■ Suffix code None
■ Initial setting 2: START-STOP
■ Example FRQ△Ø

FRQDOMAIN

FRQDOMAIN Frequency Domain Sweep

■ Function Sets whether to perform frequency lock operation of frequency axis sweep (Trace-A, B) in every sweep.

Header	Program command	Query	Response
FRQDOMAIN	FRQDOMAIN△a	FRQDOMAIN?	a

■ Value of a
LOCK: Performs a lock operation in every sweep.
UNLOCK: Performs a lock operation once in one cycle of a specified number of sweep. (lock domein sweep)

■ Suffix code None

■ Initial setting LOCK: Performs a lock operation in every sweep.

■ Example FRQDOMAIN△UNLOCK

FS

FS Full Span

■ Function Sets the frequency span to the maximum value settable in the frequency band being set.

Header	Program command	Query	Response
FS	FS	————	————

■ Example FS

FSS

FSS Frequency Step Size

■ Function Sets the frequency step size for stepping up/down the frequency (same function as SS).

Header	Program command	Query	Response
FSS	FSS△f	FSS?	FSS△f f=1 to 40000000000 Transfers the data with no suffix code in units of 1 Hz.

■ Value of f 1Hz to 40.0 GHz

■ Suffix code
None: Hz(10^0)
HZ: Hz(10^0)
KHZ,KZ: kHz(10^3)
MHZ,MZ: MHz(10^6)
GHZ,GZ: GHz(10^9)

■ Initial setting 1GHz

■ Example
FSS△1GHZ
FSS△1ØØØ

■ Restrictions according to model type and options

If equipment is MS2665C, upper limit of f is equal to 21.2 GHz.

If equipment is MS2667C, upper limit of f is equal to 30.0 GHz.

FULBAND

FULBAND EXT Mixer Band Select

■ Function

Selects an external mixer's Band. There are eleven. BANDs from 0 to 10. In GP-IB, the selection is made according to BAND NAME.

Header	Program command	Query	Response
FULBAND	FULBAND△a a=K, A, Q……J	FULBAND?	a a=K, A, Q……J

■ Value of a

Corresponds to one of K, A, Q, …, J in LIST OF EXTERNAL MIXER BANDS.

K BAND K (18.0 to 26.5 GHz, 4+)
A BAND A (26.5 to 40.0 GHz, 6+)
Q BAND Q (33.0 to 50.0 GHz, 8+)
U BAND U (40.0 to 60.0 GHz, 9+)
V BAND V (50.0 to 75.0 GHz, 11+)
E BAND E (50.0 to 90.0 GHz, 13+)
W BAND W (75.0 to 110.0 GHz, 16+)
F BAND F (90.0 to 140.0 GHz, 21+)
D BAND D (110.0 to 170.0 GHz, 26+)
G BAND G (140.0 to 220.0 GHz, 34+)
J BAND J (220.0 to 325.0 GHz, 53+)

■ Suffix code None

■ Initial setting Initial setting of a=K

■ Example

FULBAND△Q
FULBAND△J

■ Restrictions according to model type and options

This command is an MS2667C/68C dedicated command.

FUP

FUP Center Frequency Step Up

■ Function

Increases the center frequency by the frequency step size if it has been set (same function as CF△UP).

Header	Program command	Query	Response
FUP	FUP		

■ Example FUP

GATE

GATE Gate Sweep ON / OFF

■ Function Sets the gate function to be set to ON or OFF.

Header	Program command	Query	Response
GATE	GATE△sw	GATE?	sw sw=ON,OFF

■ Value of sw 1,ON: ON
Ø,OFF: OFF
■ Suffix code None
■ Initial setting OFF: OFF
■ Example GATE△ON
■ Restrictions according to model type and options
If there is no opt.06 trigger/gate circuit, this command is invalid.

GD

GD Gate Delay

■ Function Sets the delay time of the gate.

Header	Program command	Query	Response
GD	GD△t	GD?	t t=0 to 65500 Transfers the data with no suffix code in units of 1 µs.

■ Value of t 0 to 65.5ms
■ Suffix code None: ms
US: µs
MS: ms
S: s
■ Initial setting Initial value of a = 0 s
■ Example GD△2ØMS
■ Restrictions according to model type and options
If there is no opt.06 trigger/gate circuit, this command is invalid.

GDL

GDL Gate Delay

■ Function Sets the GATE delay time.

Header	Program command	Query	Response
GDL	GDL△t	GDL?	GDL△t t=0 to 65500 Transfers the data with no suffix code in units of 1 µs.

■ Value of t 0 to 65.5ms

■ Suffix code
- None: ms
- US: µs
- MS: ms
- S: s

■ Initial setting Ø: 0s

■ Example GDL△2ØMS

■ Restrictions according to model type and options

If there is no opt.06 trigger/gate circuit, this command is invalid.

GE

GE Gate End

■ Function Allows the gate interval to be terminated internally or externally.

Header	Program command	Query	Response
GE	GE△a sw=INT,EXT	GE?	a

■ Value of a
- INT: INTERNAL(Internal Timer)
- EXT: EXTERNAL(External Signal)

■ Suffix code None

■ Initial setting INT: INTERNAL(Internal Timer)

■ Example GE△INT

■ Restrictions according to model type and options

If there is no opt.06 trigger/gate circuit, this command is invalid.

GED

GED Gate End

■ Function Sets internal or external termination of the gate interval.

Header	Program command	Query	Response
GED	GED△n	GED?	GED△n

■ Value of n Ø: INTERNAL (Internal timer)
1: EXTERNAL (External signal)
■ Suffix code None
■ Initial setting Ø: INTERNAL (Internal timer)
■ Example GED△1
■ Restrictions according to model type and options
If there is no opt.06 trigger/gate circuit, this command is invalid.

GL

GL Gate Length

■ Function Sets the width of the gate.

Header	Program command	Query	Response
GL	GL△t	GL?	t t=2 to 65500 Transfers the data with no suffix code in units of 1 µs.

■ Value of t 2µs to 65.5ms
■ Suffix code None: ms
US: µs
MS: ms
S: s
■ Initial setting Initial value of t = 1 ms
■ Example GL△2ØMS
■ Restrictions according to model type and options
If there is no opt.06 trigger/gate circuit, this command is invalid.

GLN

GLN Gate Length

■ Function Sets the gate width.

Header	Program command	Query	Response
GLN	GLN△t	GLN?	GLN△t t=2 to 65500 Transfers the data with no suffix code in units of 1 µs.

■ Value of t 2µs to 65.5ms

■ Suffix code
US: µs
MS: ms
S: s

■ Restrictions according to model type and options
If there is no opt.06 trigger/gate circuit, this command is invalid.

GMD

GMD Gate Sweep On/Off

■ Function Sets the gate on or off.

Header	Program command	Query	Response
GMD	GMD△sw	GMD?	GMD△sw sw=0,1

■ Value of sw
Ø,OFF: Off
1,ON: On

■ Suffix code None

■ Initial setting Ø: Off

■ Example GMD△1

■ Restrictions according to model type and options
If there is no opt.06 trigger/gate circuit, this command is invalid.

GTOUT

GTOUT GPIB Talker time out

■ Function

Sets the time-out of the GPIB talker function (plotter/printer output, data output from PTA, etc.).
This time-out includes the sweep wait time of trigger sweeping.

Header	Program command	Query	Response
GTOUT	GTOUT△t	GTOUT?	t

■ Value of t

1 to 255: 1 to 255s
Ø: No time-out (infinite wait state)

■ Suffix code None

■ Initial setting 3Ø: 30s

■ Example GTOUT△6Ø

HN

HN Band Select

■ Function Sets the band.

Header	Program command	Query	Response
HN	HN△sw sw=0 to 5	HN?	sw sw=0 to 5 ***

■ Value of sw

		(MS2668C)
Ø:	BAND 0	BAND 0
1:	BAND 1^-	BAND 1^-
2:	BAND 1^+	BAND 1^+ (n=1)
3:	BAND 2^+	BAND 1^+ (n=2)
4:	BAND 3^+	BAND 2^- (n=4)
5:	BAND 4^+ (MS2667C)	BAND 3^- (n=6)

■ Suffix code Non6

■ Initial setting (BAND△AUTO)

■ Example HN△Ø

■ Note If there is HN△AUTO, response is "***".

■ Restrictions according to model type and options

If equipment is MS2665C, SW=5 cannot be selected.

HNLOCK

HNLOCK Band Select

■ Function Sets the band.

Header	Program command	Query	Response
HNLOCK	HNLOCK△a a=0 to 5, OFF	HNLOCK?	b

■ Value of a

Ø:	BAND 0	(Same function as BNDC△Ø)
1:	BAND 1^{-}	(Same function as BNDC△1–)
2:	BAND 1^{+}	(Same function as BNDC△1+)
3:	BAND 2^{+}	(Same function as BNDC△2+)
4:	BAND 3^{+}	(Same function as BNDC△3+)
5:	BAND 4^{+}	(Same function as BNDC△4+) (MS2667C)
OFF:	BAND AUTO	(Same function as BNDC△AUTO)

For MS2668C

Ø:	BAND 0	(Same function as BNDC Ø)
1:	BAND 1^{-}	(Same function as BNDC 1)
2:	BAND 1^{+} (n=1)	(Same function as BNDC 2)
3:	BAND 1^{+} (n=2)	(Same function as BNDC 3)
4:	BAND 2^{-} (n=4)	(Same function as BNDC 4)
5:	BAND 3^{-} (n=6)	(Same function as BNDC 5)
OFF:	BAND AUTO	(Same function as BNDC AUTO)

■ Value of b

ON:	BAND 0, 1–, 1+, 2+, 3+, 4+ (For MS2668C, 0, 1–, 1+, 1++, 2–, 3–)
OFF:	BAND AUTO

■ Suffix code None

■ Initial setting OFF: BAND AUTO

■ Example HNLOCK△2

■ Restrictions according to model type and options

If equipment is MS2665C, a=5 cannot be selected.

HNUNLK

HNUNLK Band Select

■ Function Sets the band AUTO. (Same function as BNDC△AUTO, HNLOCK△OFF)

Header	Program command	Query	Response
HNUNLK	HNUNLK	————	————

■ Example HNUNLK

HOLD

HOLD Erase Error message

■ Function Erase error message.

Header	Program command	Query	Response
HOLD	HOLD	————	————

HOLDPAUSE

HOLDPAUSE Max/Min Hold Sweep Mode

■ Function Specifies the processing (pause or continue) performed after the specified average sweeping is executed.

Header	Program command	Query	Response
HOLDPAUSE	HOLDPAUSE△a	HOLDPAUSE?	a

■ Value of a Ø,OFF: Continue (∞)
2 to 1024
■ Suffix code None
■ Initial setting Ø: Continue (∞)
■ Example HOLDPAUSE△32

INI

INI Initialize

■ Function Initializes all measurement control parameters to be initialized (same function as IP).

Header	Program command	Query	Response
INI	INI	————	————

■ Example INI

INPTRNS

INPTRNS Input impedance Transformer

■ Function Selects 75Ω Input Impedance Transformer(MA1621A).

Header	Program command	Query	Response
INPTRNS	INPTRNS△sw	INPTRNS?	sw

■ Value of sw ON: 75Ω Transformer used
OFF: 75Ω Transformer not used (50Ω)

■ Suffix code None

■ Initial setting OFF

■ Example INPTRNS△ON

INZ

INZ Input impedance

■ Function Selects input impedance.

Header	Program command	Query	Response
INZ	INZ△n	INZ?	n

■ Value of n
5Ø: 50 Ohm
75: 75 Ohm

■ Suffix code None

■ Initial setting 5Ø: 50 Ohm

■ Example INZ△75

IP

IP Initialize

■ Function nitializes all measurement control parameters to be initialized (same function as INI).

Header	Program command	Query	Response
IP	IP	————	————

■ Example IP

KSA

KSA Unit for Log Scale

■ Function Sets the of LOG scale unit to dBm (same function as UNT△0).

Header	Program command	Query	Response
KSA	KSA	————	————

■ Example KSA

KSB

KSB Unit for Log Scale

■ Function Sets the LOG scale unit to dBmV (same function as UNT△2).

Header	Program command	Query	Response
KSB	KSB	————	————

■ Example KSB

KSC

KSC Unit for Log Scale

■ Function Sets the LOG scale unit to dBuV (same function as UNT△1).

Header	Program command	Query	Response
KSC	KSC	————	————

■ Example KSC

KSD

KSD Unit for Log Scale

■ Function Sets the LOG scale unit to V (same function as UNT△3).

Header	Program command	Query	Response
KSD	KSD	————	————

■ Example KSD

KSE

KSE Title Entry

■ Function Registers the title character string (same function as TITLE).

Header	Program command	Query	Response
KSE	KSE△text	————	————

■ Value of text String of up to 32 characters enclosed by single or double quotes

■ Example
```
KSE△"MS2665C/2667C"
KSE△'SPECTRUM ANALYZER'
```

KSG

KSG Average ON

■ Function Enables averaging.

Header	Program command	Query	Response
KSG	KSG	————	————

■ Example KSG

KSH

KSH Average OFF

■ Function Disables averaging to set the mode for waveform processing to NORMAL.

Header	Program command	Query	Response
KSH	KSH	————	————

■ Example KSH

KSO

KSO Delta Marker to Span

■ Function Sets the delta marker frequency to the frequency span
(same function as MKR△6, MKSP).

Header	Program command	Query	Response
KS0	KS0	————	————

■ Example KS0

LDN

LDN Reference Level step down

■ Function Decreases the reference level by one step.

Header	Program command	Query	Response
LDN	LDN	————	————

■ Example `LDN`

LG

LG Scale

■ Function Sets the Y axis magnification and scale.

Header	Program command	Query	Response
LG	LG△l LG△a	LG?	l

■ Value of l
- `Ø:` Sets the scaling function to linear mode.
- `1:` 1dB/div (sets the scaling function to logarithmic mode)
- `2:` 2dB/div (sets the scaling function to logarithmic mode)
- `5:` 5dB/div (sets the scaling function to logarithmic mode)
- `1Ø:` 10dB/div (sets the scaling function to logarithmic mode)

■ Value of a
- `UP:` SCALE UP
- `DN:` SCALE DOWN

■ Suffix code
- None: dB/div
- `DB,DBM,DM:` dB/div

■ Initial setting `1Ø:` 10dB/div

■ Example
`LG△UP`
`LG△5DB`

LN

LN Linear Scale

■ Function Sets the Y axis scale to linear.

Header	Program command	Query	Response
LN	LN	——	——

■ Example LN

LOADEND

LOADEND Term to download PTA library.

■ Function Terminates PTA-library registration.

Header	Program command	Query	Response
LOADEND	LOADEND	——	——

■ Example LOADEND

LOADLIB

LOADLIB Load PTA Library

■ Function Loads PTA library file from memory card.

Header	Program command	Query	Response
LOADLIB	LOADLIB△a	————	————

■ Value of a PTA-library file name (alpha-numeric characters of less than 6)
■ Example LOADLIB△a

LOS

LOS Level Offset Value

■ Function Sets the offset level.

Header	Program command	Query	Response
LOS	LOS△l	LOS?	LOS△l l=–100.00 to 100.00 Transfers the data with no suffix code in units of 1 dB.

■ Value of l –100 to 100.00dB
■ Suffix code None: dB
DB: dB
■ Initial setting Ø: 0dB
■ Example LOS△2.Ø3DB

LSS

LSS Reference Level Step size(Manual)

■ Function Sets the step size (manual values) for increasing and decreasing the reference level.

Header	Program command	Query	Response
LSS	LSS△l	LSS?	LSS△l l=0.1 to 100.0 Transfers the data with no suffix code in units of 1 dB.

■ Value of l 0.1 to 100.00dB (0.01dBstep)
■ Suffix code None: dB
DB,DBM,DM: dB
■ Initial setting Value of ℓ = 1 dB
■ Example LSS△6
LSS△1Ø

LSSA

LSSA Reference Level Step Size(Auto)

■ Function Sets the step size (auto values) for increasing and decreasing the reference level during LOG SCALE operation.

Header	Program command	Query	Response
LSSA	LSSA△n	LSSA?	LSSA△n a=1,2,5,10

■ Value of n 1: 1div
2: 2div
5: 5div
1Ø: 10div
■ Suffix code None
■ Initial setting 1: 1div
■ Example LSSA△1Ø

LUP

LUP Reference Level step up

■ Function Increases the reference level by one step.

Header	Program command	Query	Response
LUP	LUP	———	———

■ Example LUP

LVO

LVO Level Offset On/Off

■ Function Sets the level offset on or off.

Header	Program command	Query	Response
LVO	LVO△sw	LVO?	LVO△sw

■ Value of sw Ø: Off
1: On
■ Suffix code None
■ Initial setting Ø: Off
■ Example LVO△1

M1

M1 Marker Mode

■ Function Turns off the marker mode (same function as MKR△2).

Header	Program command	Query	Response
M1	M1	————	————

■ Example M1

M2

M2 Marker Mode

■ Function Sets the marker mode to NORMAL mode (same function as MKR△0).

Header	Program command	Query	Response
M2	M2	————	————

■ Example M2

M3

M3 Marker Mode

■ Function Sets the marker mode to delta marker mode (same function as MKR△1).

Header	Program command	Query	Response
M3	M3	————	————

■ Example M3

MAC

MAC Marker Active

■ Function Selects the active multi-marker.

Header	Program command	Query	Response
MAC	MAC△n	MAC?	MAC△n

■ Value of n 1 to 10
■ Suffix code None
■ Initial setting 1: Marker 1
■ Example MAC△5

MADJBWLN

MADJBWLN ADJ-CH Band Line

■ Function Sets the display of the adjacent channel range line ON/OFF.

Header	Program command	Query	Response
MADJBWLN	MADJBWLN△sw	MADJBWLN?	sw

■ Value of sw OFF: OFF
ON: ON
■ Suffix code None
■ Initial setting OFF: OFF
■ Example MADJBWLN△OFF

MADJCTRLN

MADJCTRLN ADJ-CH Center Line

■ Function Sets the display of the adjacent channel center line ON/OFF.

Header	Program command	Query	Response
MADJCTRLN	MADJCTRLN△sw	MADJCTRLN?	sw

■ Value of sw OFF: OFF
ON: ON
■ Suffix code None
■ Initial setting ON: ON
■ Example MADJCTRLN△OFF

MADJGRAPH

MADJGRAPH Adjacent CH Graph

■ Function Sets the graph display function of ADJ-CH measure ON/OFF.

Header	Program command	Query	Response
MADJGRAPH	MADJGRAPH△sw	MADHGRAPH?	sw

■ Value of sw OFF: Graph display function OFF
ON: Graph display function ON
■ Suffix code None
■ Initial setting ON: Graph display function ON
■ Example MADJGRAPH△ON

MADJINBWLN

MADJINBWLN INBAND-CH Band Line

■ Function Sets the display of the inband channel range line ON/OFF.

Header	Program command	Query	Response
MADJINBWLN	MADJINBWLN△sw	MADJINBWLN?	MADJINBWLN△sw

■ Value of sw `OFF:` OFF
`ON:` ON
■ Suffix code None
■ Initial setting `OFF:` OFF
■ Example `MADJINBWLN△OFF`

MADJMOD

MADJMOD ADJ-CH Measure Method

■ Function Selects the calculation method of ADJ-CH measure.

Header	Program command	Query	Response
MADJMOD	MADJMOD△a	MADJMOD?	a

■ Value of a `MOD:` Reference=Total Power (Mod method)
`UNMD:` Reference=REF LEVEL(Un-mod method)
`INBAND:` Reference=Inband(Inband Method)
■ Suffix code None
■ Initial setting `MOD:` Reference=Total Power(Mod Method)
■ Example `MADJMOD△MOD`

MAM

MAM AM Monitor

■ Function Selects the AM voice monitor.

Header	Program command	Query	Response
MAM	MAM△sw	MAM?	MAM△sw

■ Value of sw Ø: Monitor function OFF
1: Monitor function ON
■ Suffix code None
■ Initial setting Ø: Monitor function OFF
■ Example MAM△1
■ Restrictions according to model type and options
If there is no opt.07 AM/FM demodulator, this command is invalid.

MASK

MASK Select Mask

■ Function Selects the mask data used by the mask function.

Header	Program command	Query	Response
MASK	MASK△n	MASK?	n

■ Value of n 1 to 5 (Mask No.)
■ Suffix code None
■ Initial setting 1
■ Example MASK△1

MASKLOAD

MASKLOAD Load Mask data

■ Function Reads the mask data from the external file.

Header	Program command	Query	Response
MASKLOAD	MASKLOAD△n	————	————

■ Value of n 1 to 99
■ Suffix code None
■ Example MASKLOAD△1

MASKMCL

MASKMCL Cancel Moving Value

■ Function Cancels moving value of the mask.

Header	Program command	Query	Response
MASKMCL	MASKMCL	————	————

■ Example MASKMCL

MASKMSV

MASKMSV Save Moved Mask Data

■ Function Stores the moved mask data in the original mask data area.

Header	Program command	Query	Response
MASKMSV	MASKMSV	————	————

■ Example MASKSV

MASKMVX

MASKMVX Mask Move X

■ Function Moves the mask line along the X axis.

Header	Program command	Query	Response
MASKMVX	MASKMVX△f	MASKMVX?	f f=–4000000000Hz to 4000000000Hz

■ Value of f –40.0GHz to 40.0GHz
■ Suffix code None: Hz
KHZ,KZ: KHz
MHZ,MZ: MHz
GHZ: MHz
■ Initial setting HZ
■ Example MASKMVX△1Ø6HZ
■ Restrictions according to model type and options.
If equipment is MS2665C, upper limit of f is equal to 21.2 GHz.
If equipment is MS2667C, upper limit of f is equal to 30.0 GHz.

MASKMVY

MASKMVY Mask Move Y

■ Function Moves the mask line along the Y axis.

Header	Program command	Query	Response
MASKMVY	MASKMVY△l	MASKMVY?	l

■ Value of l –200.00dB to 200.00dB
■ Suffix code None: dB
DB,DBM,DM: dB
■ Initial setting Ø: 0dB
■ Example MASKMVY△-2.5dB

MASKSAVE

MASKSAVE Save Mask data

■ Function Stores the interior mask data in the external file.

Header	Program command	Query	Response
MASKSAVE	MASKSAVE△n	————	————

■ Value of n 1 to 99
■ Suffix code None
■ Example MASKSAVE△1

MASKSLCT

MASKSLCT Mask Limit Line Select

■ Function Selects the LIMIT LINE used to evaluate the measured results using the mask functions.

Header	Program command	Query	Response
MASKSLCT	MASKSLCT△a,sw	MASKSLCT?△a	sw sw=ON,OFF

■ Value of a
UP1: Limit1 Upper
UP2: Limit2 Upper
LW1: Limit1 Lower
LW2: Limit2 Lower
■ Value of sw
Ø,OFF: Off
1,ON: On
■ Suffix code None
■ Initial setting off
■ Example MASKSLKT△UP1,ON

MBIAS

MBIAS EXT Mixer Bias

■ Function
Sets bias current of external-mixer measuring band currently slected with value of a=0 to 20.0mA (incremented by 0.1mA)

Header	Program command	Query	Response
MBIAS	MBIAS△a a=0 to 20.0	MBIAS?	a a=0 to 20.0

■ Value of a: 0 to 20.0mA
■ Suffix code: None
■ Initial setting: Initial value of a=0 (but not to be initialized)
■ Example: MBIAS△15.2
MBIAS△1.5
■ Restrictions according to model type and options.
This command is an MS2667C/68C dedicated command.

MC

MC Frequency Counter

■ Function
Turns ON/OFF the function for measuring the marker frequency during display using the counter (same function as MEAS△FREQ).

Header	Program command	Query	Response
MC	MC△sw	————	————

■ Value of sw: ON: ON
OFF: OFF
■ Suffix code: None
■ Initial setting: OFF: OFF
■ Example: MC△ON
MC△OFF

MCL Clear Multi Marker

■ Function Deletes reegistrations of all multi-markers.

Header	Program command	Query	Response
MCL	MCL	————	————

■ Example MCL

MEAS

MEAS Measure Function

■ Function Executes each item of the Measure functions when specified.

Header	Program command	Query	Response
MEAS	MEAS△data1,data2	MEAS?	data1 data1=OFF,FREQ,NOISE,OBW, ADJ,MASK,TEMP,POWER CHPWR,CN

■ Value of data1,data2

Format1:Specifies the measurement item and whether to switch it ON/OFF or execute it.

OFF:	Measurement off
FREQ, ON:	Frequency count ON
FREQ, OFF:	Frequency count OFF
NOISE, ON:	Noise calculation ON
NOISE, OFF:	Noise calculation OFF
OBW, EXE:	Executes the OBW calculation.
ADJ, EXE:	Executes the ADJ-CH calculation.
TEMP, CHECK:	Executes the template check.
MASK, CHECK:	Executes the mask check.
POWER, EXE:	Executes the burst power calculation.

Format2: Specifies the measurement item and calculation system. Then, specifies whether to switch it ON/OFF or execute it.

NOISE,ABS:	Sets the noisecalculation (Absolute method) to ON.
NOISE,CN:	Sets the noise calculation (C/N ratio method) to ON.
OBW, XDB:	Executes the OBW calculation (X dB down method).
OBW, N:	Executes the OBW calculation (N% method).
ADJ, UNMD:	Executes the ADJ-CH calculation (R: Ref Level method).
ADJ, MOD:	Executes the ADJ-CH calculation (R: Total Power method).
ADJ, INBAND:	Executes the ADJ-CH calculation (R: Inband method).
CHPWR, ON:	Channel Power calculation ON
CHPWR, OFF:	Channel Power calculation OFF

MENU

MENU Define menu

■ Function Defines the menu key (for F-key menu).

Header	Program command	Query	Response
MENU	MENU△m,text1,text2, text3,n	———	———

■ Value of m 1001 to 1200: Menu No.
■ Value of text 1 to text3
Character string (less than 1Ø characters) enclosed by single or double quotates: Menu title 1 to 3
■ Value of n 1001 to 1020: Lower menu set
■ Suffix code None
■ Example `MENU△11ØØ," Sample *","  Menu  ","",1Ø1Ø`

MENULOAD

MENULOAD Load Menu define data

■ Function Reads out the menu define data from external files.

Header	Program command	Query	Response
MENULOAD	MENULOAD△n	———	———

■ Value of n 1 to 99
■ Suffix code None
■ Example `MENULOAD△1`

MENUSAVE

MENUSAVE Save Menu define data

■ Function Stores the interior menu define data in external files.

Header	Program command	Query	Response
MENUSAVE	MENUESAVE△n	——	——

■ Value of n 1 to 99
■ Suffix code None
■ Example `MENUSAVE△1`

MENUSET

MENUSET Define menu set

■ Function Defines the menu set (one menu set).

Header	Program command	Query	Response
MENUSET	MENUSET△m,text,f1,f2,f3,f4,f5,f6,n,p1,p2	——	——

■ Value of m 1001 to 1020: Menu Set No.
■ Value of text Character string enclosed by single or double quotates: Menu Set Title
■ Value of f1 to f6 None or 1001 to 1200: Menu No. 1 to 6 corresponding to soft keys 1 to 6.
■ Value of n None or 1001 to 1020: Next page Menu Set
■ Value of p1 1 to 4: Page No.
■ Value of p2 1 to 4: Total Page
■ Suffix code None
■ Example `MENUSET△1001,"Sample Menu",1101,1102,1103,1104,1105,1106,,1,1`

MFM

MFM FM Monitor

■ Function Selects the FM voice monitor.

Header	Program command	Query	Response
MFM	MFM△sw	MFM?	MFM△sw

■ Value of sw
Ø: Monitor function OFF
1: Monitor function ON

■ Suffix code None

■ Initial setting Ø: Monitor function OFF

■ Example MFM△1

■ Restrictions according to model type and options

If there is no opt.07 AM/FM demodulator, this command is invalid.

MFR?

MFR? Multi Marker List Query (Frequency)

■ Function Reads the frequency data at the multi marker point.

Header	Program command	Query	Response
MFR?	――――――	MFR?△n	MFR△f f=–100 to 4000000000 Transfers the data with no suffix code in units of 1 Hz.

■ Value of n 1 to 10

■ Suffix code None

MHI

MHI Highest 10 (Multi Marker)

■ Function Registers the multi markers at 10 peak points starting from the highest level.

Header	Program command	Query	Response
MHI	MHI	————	————

■ Example MHI

MHM

MHM Harmonics(Multi Marker)

■ Function Registers the multi markers to the 10th harmonic max. , based on the frequency of the active marker.

Header	Program command	Query	Response
MHM	MHM	————	————

■ Example MHM

MKA?

MKA? Marker Level Read

■ Function

Reads out the level data at the marker point. At the delta marker point, the level differences are read out (same function as MKL?).

Header	Program command	Query	Response
MKA?	________	MKA?	l v w f

■ Value of l

No unit. Level data in units of 1 dB (when display unit system for marker level is dB).
Resolution is 0.01 dB.

■ Value of v

No unit. Level data in units of 1 n V (when display unit system for marker level is V).
Resolution is 0.1 nV.

■ Value of w

No unit. Level data in units of 1 μW (when display unit system for marker level is W).
Resolution is 1 aW.

■ Value of f

No unit. Frequency data in units of 1 Hz (for FM MONITOR).
Resolution is 1 Hz.

■ Example

`MKA?`

MKACT

MKACT Marker Active

■ Function Selects the active multi markers.

Header	Program command	Query	Response
MKACT	MKACT△n	MKACT?	n

■ Value of n 1 to 10 (Multi marker No.)
■ Suffix code None
■ Initial setting 1: 1
■ Example MKACT△1

MKC

MKC Frequency Counter

■ Function Turns ON/OFF the function for measuring the marker frequency during display using the counter (same function as MEAS△FREQ).

Header	Program command	Query	Response
MKC	MKC△sw	MKC?	MKC△sw

■ Value of sw Ø: OFF
1: ON
■ Suffix code None
■ Initial setting Ø: OFF
■ Example MKC△Ø
MKC△1

MKCF

MKCF Marker to CF

■ Function Sets the marker to the center frequency (same function as MKR△3, E2).

Header	Program command	Query	Response
MKCF	MKCF	———	———

■ Example MKCF

MKD

MKD Delta Marker Mode

■ Function Sets the marker mode to the delta marker mode.

Header	Program command	Query	Response
MKD	MKD	———	———

■ Example MKD

MKF?

MKF? Marker Frequency Read

■ Function Reads out the frequency or time data at the marker point. In the delta marker mode, the frequency or time differences are read out.

Header	Program command	Query	Response
MKF?	————	MKF?	f t

■ Value of f No unit, frequency data with 1 Hz unit, Resolution 0.1 Hz
■ Value of t No unit, time data with 1 µs unit, Resolution 0.1 µs
■ Example MKF?

MKFC

MKFC Frequency Counter

■ Function Turns ON/OFF the function for measuring the marker frequency during display using the counter (same function as MEAS△FREQ).

Header	Program command	Query	Response
MKFC	MKFC△sw	MKFC?	sw

■ Value of sw 1,ON : ON
Ø,OFF: OFF
■ Suffix code None
■ Initial setting Ø: OFF
■ Example MKFC△Ø
MKFC△ON

MKFCR

MKFCR Count Resolution

■ Function Selects the resolution of the frequency counter.

Header	Program command	Query	Response
MKFCR	MKFCR△f MKFCR△a	MKFCR?	f f=1,10,100,1000 Transfers data withno suffix code in units of 1 Hz.

■ Value of f
1Hz
10Hz
100Hz
1kHz

■ Value of a
UP: UP
DN: DOWN

■ Suffix code
None: Hz(10^0)
HZ: Hz(10^0)
KHZ,KZ: kHz(10^3)
MHZ,MZ: MHz(10^6)
GHZ,GZ: GHz(10^9)

■ Initial setting 1kHz

■ Example
MKFCR△1HZ
MKFCR△UP

MKL?

MKL? Marker Level Read

■ Function Reads out the level data at the marker point. In the delta marker mode, the level differences are read out.

Header	Program command	Query	Response
MKL?	————	MKL?	l v w f

■ Value of l No unit. Level data in units of 1 dB (when display unit system for marker level is dB).
Resolution is 0.01 dB.

■ Value of v No unit. Level data in units of 1 nV (when display unit system for marker level is V).
Resolution is 0.1 nV.

■ Value of w No unit. Level data in units of 1 μW (when display unit system for marker level is W).
Resolution is 1 aW.

■ Value of f No unit. Frequency data in units of 1 Hz (for FM MONITOR).
Resolution is 1 Hz.

■ Example `MKL?`

MKLFREQ

MKLFREQ Multi Marker List Freq Absolute/Relative

■ Function Sets the multi marker list frequency (hour) display to relative or in absolute values.

Header	Program command	Query	Response
MKLFREQ	MKLFREQ△a	MKLFREQ?	a

■ Value of a `ABS:` Absolute
`REL:` Relative

■ Suffix code None

■ Initial setting `ABS:` Absolute

■ Example `MKLFREQ△REL`

MKLIST

MKLIST Multi Marker List

■ Function Turns ON/OFF the multi marker list.

Header	Program command	Query	Response
MKLIST	MKLIST△sw	MKLIST?	sw sw=ON,OFF

■ Value of sw `1,ON:` ON
`Ø,OFF:` OFF
■ Suffix code None
■ Initial setting `OFF:` OFF
■ Example `MKLIST△ON`

MKLLVL

MKLLVL Multi Marker List Level Absolute/Relative

■ Function Sets the multi marker list level display to relative or absolute values.

Header	Program command	Query	Response
MKLLVL	MKLLVL△a	MKLLVL?	a

■ Value of a `ABS:` Absolute
`REL:` Relative
■ Suffix code None
■ Initial setting `ABS:` Absolute
■ Example `MKLLVL△REL`

MKMCL

MKMCL Clear Multi Marker

■ Function Clears all the registered multi markers.

Header	Program command	Query	Response
MKMCL	MKMCL	———	———

■ Example `MKMCL`

MKMFL?

MKMFL? Multi Marker All level/frequency Query

■ Function

Header	Program command	Query	Response
MKMFL?	———	MKMFL?	f1,l1,f2,l2...fn,ln

Multimarkers 1 to 10 sequentially output the frequency/time data and level data when they are ON.

fi: For Trace-A or B, the frequency, no units, and Hz units are output.
For Trace-Time, the time, no units, and 1μs units are output.

li: The following values are output according to the level data, no units, and marker level indication units:

For dB units.	Level data in 1 dB units, resolution:	0.01 dB
For V.	Level data in 1 nV units, resolution:	0.1 nV
For W.	Level data in 1 μW units, resolution:	1 aW
For FM monitors.	Frequency data in 1 Hz units, resolution:	1 Hz

MKMHI

MKMHI Multi Marker

■ Function Registers multi markers at the peak point from the maximum level down to the tenth in descending order. (HIGHEST 10)

Header	Program command	Query	Response
MKMHI	MKMHI	————	————

■ Example MKMHI

MKMHRM

MKMHRM Multi Marker

■ Function Registers multi markers at the harmonic frequency ranging from the reference active marker frequency up to the tenth. (HARMONICS)

Header	Program command	Query	Response
MKMHRM	MKMHRM	————	————

■ Example MKMHRM

MKMIN

MKMIN Minimum Search

■ Function Finds the minimum point of the spectrum being displayed and moves the marker to that point.

Header	Program command	Query	Response
MKMIN	MKMIN	————	————

■ Example MKMIN

MKML?

MKML? Multi Marker List Query (Level)

■ Function Reads out the level data at multi markers.

Header	Program command	Query	Response
MKML?	————	MKML?△n	l v w f

■ Value of n 1 to 10 (multi marker No.)

■ Value of l No unit. Level data in units of 1 dB (when display unit system for marker level is dB).
Resolution is 0.01 dB.

■ Value of v No unit. Level data in units of 1 nV (when display unit system for marker level is V).
Resolution is 0.1 nV.

■ Value of w No unit. Level data in units of 1 μW (when display unit system for marker level is W).
Resolution is 1 aW.

■ Value of f No unit. Frequency data in units of 1 Hz (for FM MONITOR).
Resolution is 1 Hz.

■ Suffix code None

MKMP

MKMP Marker Position

■ Function Specifies the frequency of a specified multi marker number.

Header	Program command	Query	Response
MKMP	MKMP△n,f	MKMP?△n	f f=–100000000 to 40000000000 Transfers the data with no suffix code in units of 1 Hz.

■ Value of n 1 to 10 (multi marker No.)
■ Value of f –100MHz to 40.0GHz
■ Suffix code None: Hz(10^0)
HZ: Hz(10^0)
KHZ,KZ: kHz(10^3)
MHZ,MZ: MHz(10^6)
GHZ,GZ: GHz(10^9)
■ Example MKMP△5,24ØØMKZ
■ Restrictions according to model type and options.
If equipment is MS2665C, upper limit of f is equal to 21.2 GHz.
If equipment is MS2667C, upper limit of f is equal to 30.0 GHz.

MKMULTI

MKMULTI Multi Marker

■ Function Turns ON/OFF the multi marker.

Header	Program command	Query	Response
MKMULTI	MKMULTI△sw	MKMULTI?	sw sw=ON,OFF

■ Value of sw 1,ON: ON
Ø,OFF: OFF
■ Suffix code None
■ Initial setting OFF: OFF
■ Example MKMULTI△ON

MKN

MKN Marker Position

■ Function Specifies the zone marker center position on the X axis in the frequency or time unit.

Header	Program command	Query	Response
MKN	MKN△f MKN△t MKN△a	MKN?	f,t f=–100000000 to 0 to 40000000000 Transfers the data with no suffix code in units of 1 Hz. t=–1000000000 to 1000000000 Transfers the data with no suffix code in units of 1 µs.

■ Value of f –100 MHz to 40.0 GHz (specified when the valid trace is A, B, or BG)

■ Value of t –1000 s to 1000 s (specified when the valid trace is TIME)

■ Value of a
UP: UP
DN: DOWN

■ Suffix code
f:
None: Hz(10^0)
HZ: Hz(10^0)
KHZ,KZ: kHz(10^3)
MHZ,MZ: MHz(10^6)
GHZ,GZ: GHz(10^9)
t:
None: ms
US: µs
MS: ms
S: s

■ Example
MKN△100MHZ
MKN△UP

■ Restrictions according to model type and options.
If equipment is MS2665C, upper limit of f is equal to 21.2 GHz.
If equipment is MS2667C, upper limit of f is equal to 30.0 GHz.

MKOFF

MKOFF Marker Mode

■ Function Turns off the marker mode.

Header	Program command	Query	Response
MKOFF	MKOFF△a	————	————

■ Value of a ALL: Marker off
None: Marker off
■ Suffix code None
■ Example MKOFF△ALL
MKOFF

MKP

MKP Marker Position

■ Function Specifies the zone marker center position on the X axis in the point unit (same function as MKZ).

Header	Program command	Query	Response
MKP	MKP△p	MKP?	p p=0 to 500

■ Value of p 0 to 500
■ Suffix code None
■ Initial setting Value of p=250
■ Example MKP△25Ø
MKP△5ØØ

MKPK

MKPK Peak Search

■ Function — Searches the spectrum being displayed for one of the special points, and moves the marker to that point.

Header	Program command	Query	Response
MKPK	MKPK△a	————	————

■ Value of a
None: SEARCH PEAK(MAX)
HI: SEARCH PEAK(MAX)
NH: SEARCH NEXT PEAK
NR: SEARCH NEXT RIGHT PEAK
NL: SEARCH NEXT LEFT PEAK

■ Suffix code — None

■ Example
MKPK△HI
MKPK△NL

MKPX

MKPX Peak Resolution(Excursion)

■ Function — Switches the marker mode and executes the 'MKR to 'functions.

Header	Program command	Query	Response
MKPX	MKPX△l	MKPX?	l l=0.01 to 50.00 Transfers the data with no suffix code in units of 1 dB.

■ Value of l — 0.01dB to 50.00dB

■ Suffix code
None: dB
DB: dB

■ Initial setting — 5.Ø: 5dB

■ Example — MKPX△1ØDB

MKR

MKR Marker Mode

■ Function Switches the marker mode and executes the 'MKR to 'functions.

Header	Program command	Query	Response
MKR	MKR△n	MKR?	MKR△n n=0 to 7

■ Value of n
Ø: NORMAL
1: DELTA
2: OFF
3: MKR to CF
4: MKR to REF
5: MKR to CF step size
6: △MKR to SPAN
7: ZONE to SPAN

■ Suffix code None

■ Initial setting Ø: NORMAL

■ Example MKR△Ø

MKRL

MKRL Marker to REF

■ Function Sets the detection resolution of the peak point.

Header	Program command	Query	Response
MKRL	MKRL	————	————

■ Example MKRL

MKS

MKS Peak Search

■ Function Searches the spectrum being displayed for one of the special points, and moves the marker to that point.

Header	Program command	Query	Response
MKS	MKS△n n=0 to 2,9 to 11	————	————

■ Value of n

Ø: SEARCH PEAK (MAX)
1: SEARCH NEXT PEAK
2: SEARCH DIP (MIN)
9: SEARCH NEXT RIGHT PEAK
1Ø: SEARCH NEXT LEFT PEAK
11: SEARCH NEXT DIP

■ Suffix code None

■ Example MKS△Ø
MKS△9

MKSLCT

MKSLCT Select Multi Marker

■ Function Selects one of the multi markers (1 to 10) and sets it to ON or OFF.

Header	Program command	Query	Response
MKSLCT	MKSLCT△n,sw	MKSLCT?△n	sw sw=ON,OFF

■ Value of n 1 to 10 (multi marker No.)

■ Value of sw

1,ON: ON
Ø,OFF: OFF

■ Suffix code None

■ Initial setting OFF: OFF

■ Example MKSLCT△3,ON

MKSP

MKSP Delta Marker to Span

■ Function Sets the delta marker frequency to the span (same function as MKR△6,KSO).

Header	Program command	Query	Response
MKSP	MKSP	———	———

■ Example MKSP

MKSRCH

MKSRCH Marker Search Mode

■ Function Sets the marker search mode.

Header	Program command	Query	Response
MKSRCH	MKSRCH△a	MKSRCH?	a

■ Value of a PEAK: Peak Marker
DIP: Dip Marker
■ Suffix code None
■ Initial setting PEAK: Peak Marker
■ Example MKSRCH△PEAK

MKSS

MKSS Marker to CF Step Size

■ Function Sets the marker frequency as the frequency step size (same function as MKR△5,E3).

Header	Program command	Query	Response
MKSS	MKSS	————	————

■ Example `MKSS`

MKTRACE

MKTRACE Active Marker Trace

■ Function Specifies the trace for displaying the marker when the display format is trace A on B.

Header	Program command	Query	Response
MKTRACE	MKTRACE△tr	MKTRACE?	tr

■ Value of tr
`TRA:` Trace A
`TRB:` Trace B

■ Suffix code None

■ Initial setting `TRA:` Trace A

■ Example `MKTRACE△TRB`

MKTRACK

MKTRACK   Tracking ON/OFF

■ Function   Sets the signal tracking function to ON/OFF.

Header	Program command	Query	Response
MKTRACK	MKTRACK△sw	MKTRACK?	sw sw=ON.OFF

■ Value of sw   1,ON: ON
Ø,OFF: OFF
■ Suffix code   None
■ Initial setting   OFF: OFF
■ Example   MKTRACK△ON

MKW

MKW   Zone Marker Width

■ Function   Specifies the zone marker width in the div unit.

Header	Program command	Query	Response
MKW	MKW△n	MKW?	MKW△n a=0 to 2,5 to 7

■ Value of n   Ø: 0.5div
1:    Spot
2:    10div
5:    1div
6:    2div
7:    5div
■ Suffix code   None
■ Initial setting   5:   1div
■ Example   MKW△1
MKW△5

MKZ

MKZ Zone Marker Position

■ Function Specifies the zone marker center position on the X axis in the point unit (same function as MKP).

Header	Program command	Query	Response
MKZ	MKZ△p	MKZ?	MKZ△p

■ Value of p 0 to 500
■ Suffix code None
■ Initial setting Value of p=250
■ Example MKZ△25Ø
MKZ△5ØØ

MKZF

MKZF Zone Marker Position

■ Function Specifies the zone marker center position on the X axis in onw od rhw frequency or time units.

Header	Program command	Query	Response
MKZF	MKZF△f MKZF△t	MKZF?	f t f=–100000000 to 0 to 40000000000 Transfers the data with no suffix code in units of 1 Hz. t=–1000000000 to 1000000000 Transfers the data with no suffix code in units of 1 µs.

■ Value of f –100 MHz to 40.0 GHz (specified when the valid trace is A, B, or BG)

■ Value of t –1000 s to 1000 s (specified when the valid trace is TIME)

■ Suffix code

f:	None:	Hz(10^0)
	HZ:	Hz(10^0)
	KHZ,KZ:	kHz(10^3)
	MHZ,MZ:	MHz(10^6)
	GHZ,GZ:	GHz(10^9)
t:	None:	ms
	US:	µs
	MS:	ms
	S:	s

■ Example

```
MKZF△1ØØMHZ
MKZF△12ØØØØØØØØ
```

■ Restrictions according to model type and options.

If equipment is MS2665C, upper limit of f is equal to 21.2 GHz.
If equipment is MS2667C, upper limit of f is equal to 30.0 GHz.

MLI

MLI Multi Marker List

■ Function Executes On/Off to the multi marker list.

Header	Program command	Query	Response
MLI	MLI△sw	MLI?	MLI△sw sw=0,1

■ Value of sw Ø,OFF: Off
1,ON: On
■ Suffix code None
■ Initial setting 1: On
■ Example MLI△Ø

MLO

MLO Multi Marker Off

■ Function Executes Off to the multi marker function.

Header	Program command	Query	Response
MLO	MLO	——	——

■ Example MLO

MLR?

MLR? Multi Marker List Query (Level)

■ Function Reads out the level data at the multi marker point.

Header	Program command	Query	Response
MLR?	————	MLR?△n	MLR△l v w f

■ Value of n 1 to 10

■ Value of l No unit. Level data in units of 1 dB (when display unit system for marker level is dB). Resolution is 0.01 dB.

■ Value of v No unit. Level data in units of 1 nV (when display unit system for marker level is V). Resolution is 0.1 nV.

■ Value of w No unit. Level data in units of 1 μW (when display unit system for marker level is W). Resolution is 1 aW.

■ Value of f No unit. Frequency data in units of 1 Hz (for FM MONITOR). Resolution is 1 Hz.

MMASK

MMASK Select Mask

■ Function Selects one of masks 1 to 5 used for mask management functions.

Header	Program command	Query	Response
MMASK	MMASK△n	MMASK?	n

■ Value of n 1 to 5 (mask No.)
■ Suffix code None
■ Initial setting 1
■ Example MMASK△1

MMASKDEL

MMASKDEL Delete MASK

■ Function Removes one point from the mask data.

Header	Program command	Query	Response
MMASKDEL	MMASKDEL△p	————	————

■ Value of p 1 to 32 (Point No.)
■ Suffix code None
■ Initial setting (None)
■ Example MMASKDEL△1Ø

MMASKDSP

MMASKDSP Mask Display Mode

■ Function Specifies how the mask management screen is displayed.

Header	Program command	Query	Response
MMASKDSP	MMASKDSP△a	MMASKDSP?	a sw=GRAPH,LIST

■ Value of a GRAPH: GRAPH
LIST: LIST
■ Suffix code None
■ Initial setting LIST
■ Example MMASKDSP△GRAPH

MMASKIN

MMASKIN Insert Point

■ Function Adds one point to the mask data.

Header	Program command	Query	Response
MMASKIN	MMASKIN△p,f,l	————	————

■ Value of p 1 to 32 (Point No.)
■ Value of f 0 to 40.0 GHz
■ Value of l 200.00dBm to 200.00dBm (ABSOLUTE)
200.00dB to 200.00dB (RELATIVE)
■ Suffix code

p:	None	
f:	None:	Hz
	Hz:	Hz
	KHZ,KZ:	KHz
	MHZ,MZ:	MHz
	GHZ:	GHz
l:	None	
	DB,DBM,DM:	dB or dBm

■ Initial setting (None)
■ Example `MMASKIN△3,1ØØMHZ,-2Ø.5DBM`

MMASKINI

MMASKINI Initiate Line / Mask

■ Function Initializes the template limit line data.

Header	Program command	Query	Response
MMASKINI	MASKINI△a	————	————

■ Value of a
UP1: LIMIT 1 UPPER
UP2: LIMIT 2 UPPER
LW1: LIMIT 1 LOWER
LW2: LIMIT 2 LOWER

■ Suffix code None

MMASKL

MMASKL Select Line

■ Function Selects the type of limit lines used for mask management functions.

Header	Program command	Query	Response
MMASKL	MMASKL△a	MMASKL?	a

■ Value of a
UP1: LIMIT 1 UPPER
UP2: LIMIT 2 UPPER
LW1: LIMIT 1 LOWER
LW2: LIMIT 2 LOWER

■ Suffix code None

MMASKLABEL

MMASKLABEL Mask Label

■ Function Specifies the mask label (name).

Header	Program command	Query	Response
MMASKLABEL	MMASKLABEL△n,text	MMASKLABEL?n	text

■ Value of n 1 to 5 (Mask No.)
■ Value of text Character string within 24 words enclosed by single or double quotes.
■ Suffix code None
■ Initial setting (None)
■ Example MMASKLABEL△1,"std-01"
MMASKLABEL△2,'CHECK01'

MMASKPD?

MMASKPD? Read Limit Line Point Data

■ Function Reads out one point of the mask data.

Header	Program command	Query	Response
MMASKPD?	————	MMASKPD?△p	f l f=–0 to 40000000000 Transfers the data with no suffix code in units of 1 Hz. l=–200.00 to 200.00 Transfers the data with no suffix code in units of 1 dB.

■ Value of p 1 to 32 (Point No.)
■ Suffix code None
■ Initial setting (None)
■ Example MMASKPD?△1

MMASKREL

MMASKREL Template Level Mode

■ Function Allows the mask level data to be set in relative or absolute values.

Header	Program command	Query	Response
MMASKREL	MMASKREL△sw	MMASKREL?	sw

■ Value of sw ON: RELATIVE
OFF: ABSOLUTE
■ Suffix code None
■ Initial setting OFF: ABSOLUTE
■ Example MMASKREL△ON

MMASKRP

MMASKRP Replace Point

■ Function Replaces one point of the mask data.

Header	Program command	Query	Response
MMASKRP	MMASKRP△p,f,1	———	———

■ Value of p 1 to 32 (Point No.)
■ Value of f 0 to 40.0GHz
■ Value of l –200.00dBm to 200.00dBm (ABSOLUTE)
–200.00dB to 200.00dB(RELATIVE)
■ Suffix code p: None
f: None: Hz
Hz: Hz
KHZ,KZ: KHz
MHZ,MZ: MHz
GHZ: GHz
l: None: dB or dBm
DB,DBM,DM: dB or dBm
■ Initial setting (None)
■ Example MMASKRP△1Ø.7MHZ,-2Ø.5DBM

MNOISE

MNOISE Noise Measure Method

■ Function Selects the calculation method for noise measurement.

Header	Program command	Query	Response
MNOISE	MNOISE△a	MNOISE?	a

■ Value of a ABS: Absolute method
CN: C/N Ratio method
■ Suffix code None
■ Initial setting ABS: Absolute method
■ Example MNOISE△ABS

MOBW

MOBW OBW Measure Method

■ Function Selects the calculation method for OBW.

Header	Program command	Query	Response
MOBW	MOBW△a	MOBW?	a

■ Value of a XDB: XdB Down method
N: N% method
■ Suffix code None
■ Initial setting N: N% method
■ Example MOBW△N

MON

MON Monitor Mode

■ Function Selects the function for monitoring the sound from the detector output.

Header	Program command	Query	Response
MON	MON△a	MON?	a

■ Value of a
- AM: Amplitude Modulation
- FM: Freqency Modulation (for broadcasting)
- FM NARROW: Narrow band FM (for communication)
- OFF: OFF

■ Suffix code None
■ Initial setting OFF: OFF
■ Example MON△AM
■ Restrictions according to model type and options
If there is no opt.07 AM/FM demodulator, this command is invalid.

MONVOL

MONVOL Monitor Volume

■ Function Adjusts the volume of the sound monitor.

Header	Program command	Query	Response
MONVOL	MONVOL△n	MONVOL?	n

■ Value of n 0 to 20 (1step)
■ Suffix code None
■ Initial setting 10
■ Example MONVOL△1Ø
■ Restrictions according to model type and options
If there is no opt.07 AM/FM demodulator, this command is invalid.

MOV

MOV Move Trace

■ Function Copies the specified trace wave data.

Header	Program command	Query	Response
MOV	MOV△tr1,tr2	————	————

■ Value of tr1,tr2 TRA: Trace-A
TRB: Trace-B
■ Suffix code None
■ Example MOV△TRA,TRB

MPS

MPS Marker Position

■ Function Specifies the position of a specified multi marker.

Header	Program command	Query	Response
MPS	MPS△n,p	MPS?△n	MPS△p

■ Value of n 1 to 10
■ Value of p Ø to 500
■ Suffix code None
■ Initial setting Ø: Left side of the wave display
■ Example MPS△1,25Ø

MSE

MSE Select Multi Marker

■ Function Sets a specified multi marker on or off.

Header	Program command	Query	Response
MSE	MSE△n,sw	MSE?△n	MSE△sw sw=0,1

■ Value of n 1 to 10
■ Value of sw Ø,OFF: Off
1,ON: On
■ Suffix code None
■ Initial setting 1,1: Marker 1: On
2 to 1Ø,Ø: Markers 2 to 10: Off
■ Example MSE△2,ON

MSOPEN

MSOPEN Open menu set

■ Function Opens a menu set. (Display)

Header	Program command	Query	Response
MSOPEN	MSOPEN△m	———	———

■ Value of m 1001 to 1020: Menu set number
■ Suffix code None
■ Example MSOPEN△1ØØ1

MTØ

MTØ Tracking OFF

■ Function Sets the signal tracking function to OFF.

Header	Program command	Query	Response
MTØ	MTØ	———	———

■ Example MTØ

MT1

MT1 Tracking ON

■ Function Sets the signal tracking function to ON.

Header	Program command	Query	Response
MT1	MT1	———	———

■ Example MT1

MTEMP

MTEMP Select Template

■ Function Selects one of templates 1 to 5 used for template management functions.

Header	Program command	Query	Response
MTEMP	MTEMP△n	MTEMP?	n

■ Value of n 1 to 5 (template No.)
■ Suffix code None
■ Initial setting 1
■ Example MTEMP△1

MTEMPDEL

MTEMPDEL Delete Template

■ Function Deletes one point of the template data.

Header	Program command	Query	Response
MTEMPDEL	MTEMPDEL△p	———	———

■ Value of p 1 to 32 (Point No.)
■ Suffix code None
■ Initial setting (None)
■ Example MTEMPDEL△1Ø

MTEMPDSP

MTEMPDSP Template Display Mode

■ Function Specifies how the template management screen is displayed.

Header	Program command	Query	Response
MTEMPDSP	MTEMPDSP△a	MTEMPDSP?	a

■ Value of a GRAPH: GRAPH
LIST: LIST
■ Suffix code None
■ Initial setting LIST
■ Example MTEMPDSP△GRAPH

MTEMPIN

MTEMPIN Insert Point

■ Function Adds one point to the template data.

Header	Program command	Query	Response
MTEMPIN	MTEMPIN△p,t,l	———	———

■ Value of p 1 to 32 (Point No.)
■ Value of t –1000 s to 1000 s
■ Value of l –200.00dBm to 200.00dBm (ABSOLUTE)
–200.00dB to 200.00dB(RELATIVE)
■ Suffix code p: None
t: None: ms
US: μs
MS: ms
S: s
l: None: dB or dBm
DB,DBM,DM: dB or dBm
■ Initial setting (None)
■ Example MTEMPIN△3.1ØMS,-2Ø.5DBM

MTEMPINI

MTEMPINI Initiate Line / Template

■ Function Initializes the template limit line data.

Header	Program command	Query	Response
MTEMPINI	MTEMPINI△a	———	———

■ Value of a
UP1: LIMIT 1 UPPER
UP2: LIMIT 2 UPPER
LW1: LIMIT 1 LOWER
LW2: LIMIT 2 LOWER

■ Suffix code None

■ Example MTEMPINI△UP1

MTEMPL

MTEMPL Select Line

■ Function Selects the type of limit lines used for template management functions.

Hea der	Program command	Query	Response
MTEMPL	MTEMPL△a	MTEMPL?	a

■ Value of a
UP1: LIMIT 1 UPPER
UP2: LIMIT 2 UPPER
LW1: LIMIT 1 LOWER
LW2: LIMIT 2 LOWER

■ Suffix code None

MTEMPLABEL

MTEMPLABEL Template Label

■ Function Specifies the template label (name).

Header	Program command	Query	Response
MTEMPLABEL	MTEMPLABEL△n,text	MTEMPLABEL?n	text

■ Value of n 1 to 5 (Template No.)
■ text Character string within 24 words enclosed by single or double quotes.
■ Suffix code None
■ Initial setting (None)
■ Example MTEMPLABEL△1,"RCR-28"
MTEMPLABEL△2,'CHECKØ1'

MTEMPPD?

MTEMPPD? Read Limit Line Point Date

■ Function Reads out one point of the template data.

Header	Program command	Query	Response
MTEMPPD?	————	MTEMPPD?△p p=1 to 32	t,l t=–1000000000 to 1000000000 Transfers the data with no suffix code in units of 1 µs. l=–200.00 to 200.00 Transfers the data with no suffix code in units of 1 dB.

■ Value of p 1 to 32 (Point No.)
■ Suffix code None
■ Initial setting (None)
■ Example MTEMPPD?△1

MTEMPREL

MTEMPREL Template Level Mode

■ Function Allows the template level data to be set in relative or absolute values.

Header	Program command	Query	Response
MTEMPREL	MTEMPREL△sw	MTEMPREL?	sw

■ Value of sw
ON: RELATIVE
OFF: ABSOLUTE

■ Suffix code None

■ Initial setting OFF ABSOLUTE

■ Example MTEMPREL△ON

MTEMPRP

MTEMPRP Replace Point

■ Function Replaces one point of the template data.

Header	Program command	Query	Response
MTEMPRP	MTEMPRP△p,t,l	———	———

■ Value of p 1 to 32 (Point No.)

■ Value of t –1000 to 1000s

■ Value of l
–200.00 to 200.00dBm (ABSOLUTE)
–200.00 to 200.00dB (RELATIVE)

■ Suffix code

p:	None	
t:	None:	ms
	US:	μs
	MS:	ms
	S:	s
l:	None:	dB or dBm
	DB,DBM,DM:	dB or dBm

■ Initial setting None

■ Example MTEMPRP△3.1ØMS,-2Ø.5DBM

MVL

MVL Monitor volume

■ Function Adjusts the volume of the sound monitor.

Header	Program command	Query	Response
MVL	MVL△n	MVL?	MVL△n

■ Value of n 0 to 20
■ Suffix code None
■ Initial setting 1Ø
■ Example MVL△5
■ Restrictions according to model type and options
If there is no opt.07 AM/FM demodulator, this command is invalid.

MXMH

MXMH Max Hold

■ Function Sets the mode for processing the trace waveform to MAX HOLD.

Header	Program command	Query	Response
MXMH	MXMH△tr	———	———

■ Value of tr TRA: Trace A
TRA: Trace B
■ Suffix code None
■ Example MXMH△TRA

MXRMODE

MXRMODE INT/EXT Mixer Band Select

■ Function Selects either internal mixer BAND or external mixer BAND.

Header	Program command	Query	Response
MXRMODE	MXRMODE△a a=INT, EXT	MXRMODE?	a a=INT, EXT

■ Value of a INT: INTERNAL MIXER
EXT: EXTERNAL MIXER
■ Suffix code None
■ Initial setting INT: INTERNAL MIXER
■ Example MXRMODE△0
MXRMODE△1
■ Restrictions according to model type and options
This command is an MS2667C/68C dedicated command.

MZW

MZW Zone Marker Width

■ Function Specifies the zone marker width on the X axis in the point unit.

Header	Program command	Query	Response
MZW	MZW△p	MZW?	MZW△p

■ Value of p 1 to 501
■ Suffix code None
■ Initial setting w=51
■ Example MZW△1
MZW△51
MZW△5Ø1

MZWF

MZWF Zone Marker Width

■ Function Specifies the zone marker width on the X axis in one of the frequency units.

Header	Program command	Query	Response
MZWF	MZWF△f	MZWF?	f f=1 to 40000000000 Transfers the data with no suffix code in units of 1 Hz

■ Value of f 1Hz to 40.0 GHz

■ Suffix code
None: Hz(10^0)
HZ: Hz(10^0)
KHZ,KZ: kHz(10^3)
MHZ,MA: MHz(10^6)
GHZ,GZ: GHz(10^9)

■ Initial setting Width equivalent to 1 div (MS2665C: 2.12 GHz, MS2667C: 3 GHz, MS2668C: 4 GHz)

■ Example
MZWF△1ØØ
MZWF△1MHZ

■ Restrictions according to model type and options
If equipment is MS2665C, upper limit of f equal to 21.2 GHz.
If equipment is MS2667C, upper limit of f equal to 30.0 GHz.

OBWN

OBWN OBW N% Value

■ Function Sets the conditions of the occupied frequency bandwidth in units of 1%.

Header	Program command	Query	Response
OBWN	OBWN△n	OBWN?	n

■ Value of n 0.01 to 99.99 (0.01 step) : 0.01 to 99.99% (0.01%step)
■ Suffix code None
■ Initial setting 99%
■ Example OBWN△8Ø

OBWXDB

OBWXDB OBW XdB Value

■ Function Sets the conditions of the occupied frequency bandwidth in units of 1 dB.

Header	Program command	Query	Response
OBWXDB	OBWXDB△l	OBWXDB?	l

■ Value of l 0.01 to 100 (0.01 step) : 0.01 to 100dB (0.01dB step)
■ Suffix code None: dB
DB: dB
■ Initial setting 25dB
■ Example OBWXDB△6DB

PARADSP

PARADSP Parameter display type

■ Function Sets the display method for the parameter type.

Header	Program command	Query	Response
PARADSP	PARADSP△n	PARADSP?	n

■ Value of n
1: TYPE1 (Displays the title and the coupled parameter)
2: TYPE2 (Displays the marker in large characters and the coupled parameter)
3: TYPE3 (Displays the marker in large characters and the title)

■ Suffix code None

■ Initial setting 1: TYPE1

■ Example PARADSP△3

PCF

PCF Peak to Center Frequency

■ Function Finds the maximum point of the spectrum being displayed, and sets the center frequency to that point.

Header	Program command	Query	Response
PCF	PCF	———	———

■ Example PCF

PINI

PINI Partial Preset

■ Function Executes partial initialization.

Header	Program command	Query	Response
PINI	PLNI△n	————	————

■ Value of n
0: Preset All (initializes all parameters in the same way as "IP" and "INI.")
1: Preset Sweep Control (initializes sweep control items.)
2: Preset Trace Parameter (initializes trace items.)
3: Preset Level Parameter (initializes vertical-axis items.)
4: Preset Freq/Time parameter (initializes horizontal-axis items.)

■ Example `PINI△Ø`

PLF

PLF Plotting Paper Form

■ Function Specifies the paper size for the plotter.

Header	Program command	Query	Response
PLF	PLF△n	PLF?	PLF△n

■ Value of n
Ø: A4
1: A3

■ Suffix code None

■ Initial setting Ø: A4

■ Example `PLF△1`

PLI

PLI Direct Plot Output Item For Plotter

■ Function: Specifies the information (e.g. waveform only, scale only) to be plotted directly.

Header	Program command	Query	Response
PLI	PLI△n	PLI?	PLI△n

■ Value of n
Ø: ALL
1: TRACE ONLY
2: SCALE ONLY

■ Suffix code: None

■ Initial setting: Ø: ALL (provided the already set is not initialized)

■ Example: PLI△Ø

PLOT

PLOT Direct Plot

■ Function: Executes direct plotting.

Header	Program command	Query	Response
PLOT	PLOT	————	————

■ Example: PLOT

PLS

PLS Direct Plot Start

■ Function Starts direct plotting.

Header	Program command	Query	Response
PLS	PLS△Ø	————	————

■ Example PLS△Ø

■ Note: This command starts the next command processing after completion of the editing print data.
To wait the next command until end of the printing, use the PRINT or PLOT command.

PLTA

PLTA Direct Plot Plotter Address

■ Function Sets the GPIB address of the plotter for direct plotting.

Header	Program command	Query	Response
PLTA	PLTA△n	PLTA?	PLTA△n

■ Value of n 0 to 30

■ Suffix code None

■ Initial setting a = 18 (provided the GPIB address already allocated is not initialized)

■ Example PLTA△Ø

PLTARA

PLTARA Plotting Size

■ Function Specifies the size of the plotting area.

Header	Program command	Query	Response
PLTARA	PLTARA△a	PLTARA?	a

■ Value of a FULL: total
QTR: 1/4 size
■ Suffix code None
■ Initial setting FULL: total
■ Example PLTARA△QTR

PLTHOME

PLTHOME Set Home Position

■ Function Initializes the printing position to the upper left-corner when the selected LOCATION is AUTO.

Header	Program command	Query	Response
PLTHOME	PLTHOME	———	———

PMCS

PMCS Memory Card

■ Function Selects the slot from the build-in memory card.

Header	Program command	Query	Response
PMCS	PMCS△a	PMCS?	a

■ Value of a
SLOT1: Slot 1 (top slot)
SLOT2: Slot 2 (bottom slot)
■ Suffix code None
■ Initial setting SLOT1: Slot 1 (provided the already set is not initialized)
■ Example PMCS△SLOT2

PMOD

PMOD Printer Type

■ Function Selects the type of printer for direct plotting.

Header	Program command	Query	Response
PMOD	PMOD△n	PMOD?	PMOD△n

■ Value of n
Ø: Printer .... HP-GL
1: Printer .... GP-GL
2: Printer .... VP-600 (ESC/P)
3: Printer .... HP2225 (Hewlett Packard)
4: BMP-format file
■ Suffix code None
■ Initial setting 2: Printer VP600
■ Example
PMOD△2
PMOD△4

PMY

PMY Dual-Port Memory

■ Function — Writes to the dual port memory or reads from the momory for PTA.
32 bytes × 32 memories

Header	Program command	Query	Response
PMY	PMY△n,b n=0 to 31 b=date	PMY?△n,c	b

■Value of n — Dual port number: 0 to 31
■ Value of b — Data enclosed in single or double quotes
■ Value of c — Number of data items read from the dual port memory: 1 to 32
■ Example — PMY△Ø,"5Ø"
PMY△Ø,1

PORT

PORT Control Port Select

■ Function — Selects the port for the external device controlled form the PTA.

Header	Program command	Query	Response
PORT	PORT△n	PORT?	PORT△n

■ Value of n
1: RS232C
2: GPIB
3: PARALLEL(CENTRO)

■ Suffix code — None
■ Initial setting — 1: RS232C (provided the already set is not initialized)
■ Example — PORT△1

POWERON

POWERON Power on State

■ Function Sets the power on status.

Header	Program command	Query	Response
POWERON	POWERON△a	POWERON?	a

■ Value of a
IP: Initialized (Preset) status
LAST: Status at last power-off
1 to 12: Reads and sets the specified recall memory contents.

■ Suffix code None
■ Initial setting LAST: Status at power-off
■ Example POWERON△12

PP

PP Presel Auto

■ Function Sets the auto tune of preselect

Header	Program command	Query	Response
PP	PP	———	———

■ Example PP

PRESEL

PRESEL Presel Tune

■ Function Sets the auto tune of preselect

Header	Program command	Query	Response
PRESEL	PRESEL△a	PRESEL?	a a= –128 to 127

■ Value of a AUTO: Auto tune
–128 to 127: MANUAL set
■ Suffix code None
■ Initial setting Ø(MANUAL) (the preselect tune already registered is not initialozed)
■ Example PRESEL△AUTO

PRIA

PRIA Direct Plot Printer Address

■ Function Sets the GPIB address of the printer for direct plotting.

Header	Program command	Query	Response
PRIA	PRIA△n	PRIA?	n

■ Value of n 0 to 30
■ Suffix code None
■ Initial setting a = 17 (provided the address already allocated is not initialized)
■ Example PRIA△17

PRINT

PRINT Direct Plot

■ Function Executes direct plotting.

Header	Program command	Query	Response
PRINT	PRINT	————	————

■ Example `PRINT`

PRINTMAG

PRINTMAG Printer Magnification

■ Function Selects printer magnification.

Header	Program command	Query	Response
PRINTMAG	PRINTMAG△a	PRINTMAG?	a

■ Value of a
`11:` 1 x 1 (Same size)
`21:` 2 x 1 (double height)
`12:` 1 x 2 (double width)
`22:` 2 x 2 (Four times)
`23:` 2 x 3 (Six times)
`24:` 2 x 4 (Eight time)

■ Suffix code None

■ Initial setting `11:` 1 x 1 (Same size)

■ Example `PRINTMAG△22`

PRL

PRL Peak to Reference Level

■ Function Finds the maximum point of the spectrum being displayed, and sets it level to the reference level.

Header	Program command	Query	Response
PRL	PRL	————	————

■ Example PRL

PRTPORT

PRTPORT Printer port

■ Function Printer port.

Header	Program command	Query	Response
PRTPORT	PRTPORT△a	PRTPORT?	a

■ Value of a
RS232C: RS232C
GPIB: GPIB
PARALLEL: PARALLEL(CENTRO)
NONE: NONE

■ Example PRTPORT△PARALLEL

■ Restrictions according to model type and options.
If there is no opt. 10 CENTRONICS INTERFACE, a=PARALLEL can not be selected.
If there is opt. 10 CENTRONICS INTERFACE, a=GPIB cannot be selected.

PRTY

PRTY Parity

■ Function Sets the parity bit for RS-232C.

Header	Program command	Query	Response
PRTY	PRTY△n	PRTY?	n

■ Value of n EVEN: Even
ODD: Odd
OFF: Off (None)
■ Suffix code None
■ Initial setting OFF: Off (None)
■ Example PRTY△EVEN

PSW

PSW Zone Sweep

■ Function Sets the zone sweep to ON/OFF.

Header	Program command	Query	Response
PSW	PSW△sw	PSW?	PSW△sw sw=ON,OFF

■ Value of sw 1,ON: ON
Ø,OFF: OFF
■ Suffix code None
■ Initial setting OFF: OFF
■ Example PSW△ON

PTA

PTA PTA Switch / PTA Status

■ Function: Sets the PTA to ON/OFF.
Reads whether PTA is BUSY or READY. (PTA OFF resets the PTA program.)

Header	Program command	Query	Response
PTA	PTA△sw	PTA?	PTA△b

■ Value of sw
1,ON: ON
Ø,OFF: OFF

■ Value of b
Ø: PTA is of Ready state.
1: PTA is of Break state.
2: PTA is of Busy state.
3: PTA is of Run state.

■ Suffix code: None

■ Initial setting: OFF: OFF (provided that PTA OFF is not affected by the INI command)

■ Example: PTA△0

PTL

PTL PTL I / O Mode

■ Function: Selects the mode for controlling PTA via GPIB/RS-232C.

Header	Program command	Query	Response
PTL	PTL△sw	PTL?	text

■ Value of sw
Ø: PTA is not controlled by GPIB/RS-232C.
1: PTA is controlled by GPIB/RS-232C.

■ Text: Text at one statement of PTA-program/PTA-library

■ Suffix code: None

■ Initial setting: OFF (provided the mode already allocated is not initialized)

■ Example
PTL△Ø: OFF
PTL△1: Input (mode to transfer a command or statement to PTA)
PTL?: Output (mode to transfer a statement from PTA to an external device)

PWRSTART

PWRSTART Power Measure Start Point

■ Function Specifies the point at which to start burst-power measurement.

Header	Program command	Query	Response
PWRSTART	PWRSTART△p	PWRSTART?	p

■ Value of p 0 to 500
■ Suffix code None
■ Initial setting 1ØØpoint
■ Example PWRSTART△1ØØ

PWRSTOP

PWRSTOP Power Measure Stop Point

■ Function Specifies the point at which to terminate burst-power measurement.

Header	Program command	Query	Response
PWRSTOP	PWRSTOP△p	PWRSTOP?	p

■ Valur of p 0 to 500
■ Suffix code None
■ Initial setting 4ØØpoint
■ Example PWRSTOP△4ØØ

RB

RB Resolution Bandwidth

■ Function Sets the resolution bandwidth (same function as RBW).

Header	Program command	Query	Response
RB	RB△f RB△a	RB?	f f=10 to 3000000 Transfers the data with no suffix code in units of 1 Hz

■ Value of f 10 Hz to 3 MHz (1/3 sequence)

■ Value of a
- UP: RBW UP
- DN: RBW DOWN
- AUTO: RBW AUTO

■ Suffix code
- f: None: Hz(10^0)
  - HZ: Hz(10^0)
  - KHZ,KZ: kHz(10^3)
  - MHZ,MZ: MHz(10^6)
  - GHZ,GZ: GHz(10^9)
- a: None

■ Initial setting RBW=calculated value when AUTO is selected for RBW

■ Example RB△3KHZ

■ Restrictions according to model type and options
- If there is no opt.02 narrow RBW; 30 Hz, 100 Hz and 300 Hz cannot be selected.
- If there is no opt.03 narrow RBW; 10 Hz, 30 Hz, 100 Hz, 300 Hz cannot be selected.

RBR

RBR Resolution Bandwidth/Span Ratio

■ Function Sets the RBW/Span Ratio.

Header	Program command	Query	Response
RBR	RBR△f	RBR?	f

■ Value of f 0.001 to 0.100 (resolution 0.001)
■ Suffix code None
■ Initial setting 0.01
■ Example RBR△0.05

RBSPAN

RBSPAN Resolution Bandwidth/Span

■ Function Sets the RBW according to RBW/Span Ratio.

Header	Program command	Query	Response
RBSPAN	RBSPAN△sw	RBSPAN?	sw

■ Value of sw OFF: OFF
0: OFF
ON: ON
1: ON
■ Initial setting OFF: OFF
■ Suffix code None
■ Example RBSPAN△ON

RBW

RBW Resolution Bandwidth

■ Function Sets the resolution bandwidth.

Header	Program command	Query	Response
RBW	RBW△n	RBW?	RBW△n

■ Value of n

Ø:	30Hz
1:	100Hz
2:	300Hz
3:	1kHz
4:	3kHz
5:	10kHz
6:	30kHz
7:	100kHz
8:	300kHz
9:	1MHz
13:	10Hz
14:	3MHz

■ Suffix code None

■ Initial setting Calculated value when AUTO is selected for RBW

■ Example RBW△5

■ Restrictions according to model type and options

- If there is no opt.02 narrow RBW, n=0, 1, 2 cannot be selected.
- If there is no opt.03 narrow RBW, n=0, 1, 2, 13 cannot be selected.

RC

RC Recall Data from Internal Register

■ Function Recalls trace data/parameter data from the built-in memory (same function as RGRC).

Header	Program command	Query	Response
RC	RC△n	————	————

■ Value of n 1 to 12 (Register No.)
■ Suffix code None
■ Example `RC△1`

RCM

RCM Recall Data from Memory Card

■ Function Recalls the measurement conditions (parameters) and measured results (traces) from memory card.

Header	Program command	Query	Response
RCM	RCM△n	————	————

■ Value of n 1 to 99 (File No.)
■ Suffix code None
■ Example `RCM△2 RCM△17`

RCS

RCS Write Off Recall Data

■ Function Recalls data from memory card and sets the storage mode to "View".

Header	Program command	Query	Response
RCS	RCS△n	————	————

■ Value of n 1 to 99
■ Suffix code None
■ Example RCS△1

RDATA

RDATA Recalled Data

■ Function Specifies the data to be recalled.

Header	Program command	Query	Response
RDATA	RDATA△a	RDATA?	a

■ Value of a
TP: Trace & Parameter
P: Parameter Only
TPV: Trace & Parameter (view)
PER: Parameter (except RLV)

■ Suffix code None
■ Initial setting TP: Trace & Parameter (provided the already set is not initialized)
■ Example RDATA△TP

RES?

RES? Measure Result

■ Function Reads out the results functions.

Header	Program command	Query	Response
RES?	————	RES?	data1 data1,data2 data1,data2,data3,data4

■ Values of data1,data2,data3, and data4

Measure control item (corresponding command)	Response	Value of data1	Value of data2	Value of data3	Value of data4
When the measure item or sub item is OFF	OFF	Not transferred	Not transferred	————	————
FREQ COUNT (MEAS△FREQ,ON)	f	Value of f with no suffix code in units of 1 Hz Resolution: 1 Hz	————	————	————
NOISE MEASURE (MEAS△NOISE,ABS) (MEAS△NOISE,C/N)	l	Value of l with no suffix code in units of 1 dB (dBm/ch, dBm/Hz, dBc/ch, dBc/Hz). Resolution: 0.01 dB	————	————	————
OBW MEASURE (MEAS△OBW,XDB) (MEAS△OBW,N)	f1,f2	Occupied bandwidth of f1 with no suffix code in units of 1 Hz. Resolution: 1 Hz	Center frequency of f2 with no suffix code in units of 1 Hz. Resolution: 1 Hz	————	————
ADJ CH MEASURE (MEAS△ADJ,UNMD) (MEAS△ADJ,MOD)	lL1,lU1 lL2,lU2	Lower channel of CHSEPA1 of lL1 with no suffix code in units of 1 dB. Resolution: 0.01 dB	Upper channel fo CH SEPA2 of lU1 with no suffix code in units of 1 dB. Resolution: 0.01 dB	Lower channel of CH SEPA2 of lL2 with no suffix code in units of 1 dB. Resolution: 0.01 dB	Upper channel of CH SEPA2 of lU2 with no suffix code in units of 1 dB. Resolution: 0.01 dB
MASK (MEAS△MASK,CHECK)	C1,C2	Value of C1(Limit 1 check result) 0:PASS1, 1:FAIL	Value of C2(Limit 2 check result) 0:PASS1, 1:FAIL	————	————
TEMPLATE (MEAS△TEMP,CHECK)	C1,C2	Value of C1(Limit 1 check result) 0:PASS1, 1:FAIL	Value of C2(Limit 2 check result) 0:PASS1, 1:FAIL	————	————
BURST POWER MEASURE (MEAS△POWER,EXE)	l,w	dB m value of l with no suffix code in units of 1 dBm. Resolution: 0.01 dB	pW value of w with no suffix code in units of 1 pW. Resolution: 1 pW	————	————
CHANNEL POWER MEASURE (MEAS△CHPWR,ON)	l1,l2 (In case of Marker not spot mode)	Value of l1 with no suffix code in units of 1 dBm. Resolution: 0.01 dB	Value of l2 with no suffix code in units of 1 dBm/Hz. Resolution: 0.01 dB	————	————
	l (In case of Marker spot mode)	Value of l with no suffix code in units of 1 dBm/Hz Resolution: 0.01 dB			

If the MEASURE function has caused a calculation error or execution error, the affected value is represented by "***".

■ Example RES?

RGRC

RGRC Recall Data from Internal Register

■ Function Recalls trace data/parameter data from the built-in register (same function as RC).

Header	Program command	Query	Response
RGRC	RGRC△n	————	————

■ Value of n 1 to 12 (Register No.)
■ Suffix code None
■ Example RGRC△1

RGSV

RGSV Save Data into Internal Register

■ Function Saves trace data/parameter data to the built-in register (same function as SV).

Header	Program command	Query	Response
RGSV	RGSV△n	————	————

■ Value of n 1 to 12 (Register No.)
■ Suffix code None
■ Example RGSV△1

RL

RL Reference Level

■ Function Sets the reference level (same function as RLV).

Header	Program command	Query	Response
RL	RL△l RL△a	RL?	l l: No units value depending on the current scalunit. the μV units are selected for V-unit system, and μW units are selected for W-unit system.

■ Value of l Value from –100 dBm to +30 dBm (0.01 dB step)

■ Value of a
UP: LEVEL STEP UP
DN: LEVEL STEP DOWN

■ Suffix code
None: No units value depending on the current scale unit. The V units are always selected when in LIN mode.
DB,DBM,DM: dBm
DBMV: dBmV
DBUV: dBμV
DBUVE: dBμV(emf)
DBUVM: dBμV/m
V: V
MV: mV
UV: μV
W: W
MW: mW
UW: μW
NW: nW
PW: pW
FW: fW

■ Initial setting l = -10 dBm

■ Example
RL△-1ØØDBM
RL△5V
RL△-1ØV
RL△UP

RLN

RLN Reference Line

■ Function Specifies the location of the data display standard line obtained using the A-B function.

Header	Program command	Query	Response
RLN	RLN△n	RLN?	RLN△n

■ Value of n
Ø: Top
1: Middle
2: Bottom

■ Suffix code None

■ Initial setting 1: Middle

■ Example RLN△2

RLV

RLV Reference Level

■ Function — Sets the reference level (same function as RL).

Header	Program command	Query	Response
RLV	RLV△l	RLV?	RLV△l l: No units value depending on the current scale unit.The µV units are selected for V-unit system, and µW unitsare selected for W-unit system.

■ Value of l — Value from –100 dBm to +30 dBm (0.01 dB step)

UP: LEVEL STEP UP
DN: LEVEL STEP DOWN

■ Suffix code

None: No units value depending on the current scale unit. The V units arealways selected when in LIN mode.
DB,DBM,DM: dBm
DBMV: dBmV
DBUV: dBµV
DBUVE: dBµV(emf)
DBUVM: dBµV/m
V: V
MV: mV
UV: µV
W: W
MW: mW
UW: µW
NW: nW
PW: pW
FW: fW

■ Initial setting — l= –10 dBm

■ Example

```
RL△-1ØØDBM
RL△5V
RL△-1ØV
```

RMK?

RMK? Reference Marker Position

■ Function Reads out the position of the reference marker.

Header	Program command	Query	Response
RMK?	―――	RMK?	RMK△a

■ Value of a 0 to 500
■ Example RMK?

ROFFSET

ROFFSET Ref. Level Offset

■ Function Turns the reference level offset ON/OFF, and sets the offset value.

Header	Program command	Query	Response
ROFFSET	ROFFSET△sw ROFFSET△l	ROFFSET?	OFF l

■ Value of sw ON: ON
OFF: OFF
■ Value of l –100.00dB to +100.00dB(0.01dB step)
■ Suffix code None: dB
DB,DBM,DM: dB
■ Initial setting Ø: 0dB
■ Example ROFFSET△OFF
ROFFSET△2ØDB

S1

S1 Sweep Mode (Continuous)

■ Function Sets the sweep mode to CONTINUOUS (same function as CONTS).

Header	Program command	Query	Response
S1	S1	——	——

■ Example S1

S2

S2 Sweep Mode (Single)

■ Function Sets the sweep mode to SINGLE (same function as SNGLS).

Header	Program command	Query	Response
S2	S2	——	——

■ Example S2

SAVELIB

SAVELIB Save PTA Library file

■ Function Saves PTA library file with extention of .LIB at memory card.

Header	Program command	Query	Response
SAVELIB	SAVELIB△a[,lib1,lib2,••]	————	————

■ Value of a PTA-library file name (alpha-numeric characters of less than 6)
■ lib1~ PTA-library name (When omitted, all the currently loaded PTA libraries are saved.)
■ Example SAVELIB△ABC,PLIB1,PLIB2
Library programs PLIB1 and PLIB2 are saved at ABC.LIB file.

SCL

SCL Log/ Linear Scale

■ Function Sets the Y axis magnification of the LOG/LIN scale.

Header	Program command	Query	Response
SCL	SCL△n	SCL△	SCL△n

■ Value of n

Ø:	1dB/div(LOG SCALE)
1:	2dB/div(LOG SCALE)
2:	5dB/div(LOG SCALE)
3:	10dB/div(LOG SCALE)
4:	1%/dev(LIN SCALE)
5:	2%/dev(LIN SCALE)
6:	5%/dev(LIN SCALE)
7:	10%/dev(LIN SCALE)

■ Suffix code None
■ Initial setting 3: 10dB/div (LOG SCALE)
■ Example SCL△Ø
SCL△5

SCR

SCR Scroll

■ Function Scrolls the displayed spectrum to the right or left by the specified scroll amount.

Header	Program command	Query	Response
SCR	SCR△a	————	————

■ Value of a
- Ø: SCROLL LEFT
- LEFT: SCROLL LEFT
- 1: SCROLL RIGHT
- RIGHT: SCROLL RIGHT

■ Suffix code None

■ Example
SCR△Ø
SCR△RIGHT

SIGID

SIGID Signal Identifier

■ Function Turns ON/OFF the sweep to distinguish actual signals to be measured from image signals when an external mixer is used.
Switches over the polarity of the band specified by EXT MIXER BAND CONTROL (FULBAND command) alternately from the + side to the –side or vice versa (e. g. from 2+ to 2–) and displays it in a swept manner.

Header	Program command	Query	Response
SIGID	SIGID△a	SIGID?	a a=0, 1

■ Value of a
- Ø, OFF: OFF
- 1, ON: ON

■ Suffix code None

■ Initial setting Ø: OFF

■ Example
SIGID△Ø
SIGID△1

■ Restrictions according to model type and options
This command is an MS2667C/68C dedicated command.

SNGLS

SNGLS Single Sweep Mode

■ Function Sets the sweep mode to single sweep (same function as S2).

Header	Program command	Query	Response
SNGLS	SNGLS	——	——

■ Example `SNGLS`

SOF

SOF Stop Frequency

■ Function Sets the stop frequency (same function as FB).

Header	Program command	Query	Response
SOF	SOF△f	SOF?	SOF△f f=-100000000 to 0 to 40000000000 Transfers the data with no suffix code in units of 1 Hz.

■ Value of f –100MHz to 40.0 GHz
■ Suffix code
None: Hz(10^0)
`HZ:` Hz(10^0)
`KHZ,KZ:` kHz(10^3)
`MHZ,MA:` MHz(10^6)
`GHZ,GZ:` GHz(10^9)
■ Initial setting `f`= 21.2GHz (MS2665C), 30.0 GHz (MS2667C), 40.0 GHz (MS2668C)
■ Example
`SOF△123MHZ`
`SOF△45.6KHZ`
■ Restrictions according to modeltype and options.
If equipment is MS2665C, upper limit of f is equal to 21.2 GHz.
If equipment is MS2667C, upper limit of f is equal to 30.0 GHz.

SP

SP Frequency Span

■ Function Sets the frequency span (same function as SPF).

Header	Program command	Query	Response
SP	SP△f SP△a	SP?	f f=0 to 40100000000 Transfers the data with no suffix code in units of 1 Hz.

■ Value of f 0Hz to 40.1 GHz
■ Value of a UP: FREQ SPAN STEP UP (same function as SPU)
DN: FREQ SPAN STEP DOWN(same function as SPD)
■ Suffix code None: Hz(10^0)
HZ: Hz(10^0)
KHZ,KZ: kHz(10^3)
MHZ,MZ: MHz(10^6)
GHZ,GZ: GHz(10^9)
■ Initial setting f= 21.2 GHz (MS2665C), 30.0 GHz (MS2667C), 40.0 GHz (MS2668C)
■ Example SP△1GHZ
■ Restrictions according to modeltype and options.
If equipment is MS2665C, upper limit of f is equal to 21.3 GHz.
If equipment is MS2667C, upper limit of f is equal to 30.1 GHz.

SPD

SPD Frequency Span Step Down

■ Function Decreases the frequency span in the 5/2/1 steps (same function as SP△DN).

Header	Program command	Query	Response
SPD	SPD	———	———

■ Example SPD

SPF

SPF Frequency Span

■ Function Sets the frequency span (same function as SP).

Header	Program command	Query	Response
SPF	SPF△f	SPF?	SPF△f f=-0 to 40100000000 Transfers the data with no suffix code in units of 1 Hz.

■ Value of f 0Hz to 40.1GHz
■ Suffix code None: Hz(10^0)
HZ: Hz(10^0)
KHZ,KZ: kHz(10^3)
MHZ,MZ: MHz(10^6)
GHZ,GZ: GHz(10^9)
■ Initial setting f= 21.2 GHz (MS2665C), 30.0 GHz (MS2667C), 40.0 GHz (MS2668C)
■ Example SPF△1Ø1MHZ
SPF△1.5GHZ
■ Restrictions according to modeltype and options.
If equipment is MS2665C, upper limit of f is equal to 21.3 GHz.
If equipment is MS2667C, upper limit of f is equal to 30.1 GHz.

SPFUNC

SPFUNC FM Monitor

■ Function Sets the function for monitoring the trace time waveform.

Header	Program command	Query	Response
SPFUNC	SPFUNC△sw	SPFUNC?	sw

■ Value of sw OFF: OFF
FM: FM MONITOR
■ Suffix code None
■ Initial setting OFF: OFF
■ Example SPFUNC△FM

SPU

SPU Frequency Span Step. Up

■ Function Increases the frequency span in the 1/2/5 steps (same function as SP△UP).

Header	Program command	Query	Response
SPU	SPU	————	————

■ Example SPU

SRCHTH

SRCHTH Peak Search Threshold

■ Function Sets the threshold function for detecting a peak point.

Header	Program command	Query	Response
SRCHTH	SRCHTH△a	SRCHTH?	sw sw=OFF,ABOVE,BELOW

■ Value of sw Ø,OFF: No threshold function
1,ON: Threshold function
■ Value of a ABOVE: Above detection
BELOW: Below detection
■ Suffix code None
■ Initial setting OFF: No threshold function
■ Example SRCHTH△ABOVE

SRCNORM

SRCNORM Normalize

■ Function Selects the ON/OFF of the nolmalizing processing(A–B+DL–>A).

Header	Program command	Query	Response
SRCNORM	SRCNORM△sw	SRCNORM?	sw sw=ON,OFF

■ Value of sw ON: on
OFF: off
■ Suffix code None
■ Initial setting OFF: off
■ Example SRCNORM△ON

SS

SS Frequency Step Size

■ Function Sets the frequency step size for stepping up/down the frequency (same function as FSS).

Header	Program command	Query	Response
SS	SS△f	SS?	f f=1to 40000000000 Transfers the data with no suffix code in units of 1 Hz.

■ Value of f 1Hz to 40.0 GHz
■ Suffix code None: Hz(10^0)
HZ: Hz(10^0)
KHZ,KZ: kHz(10^3)
MHZ,MZ: MHz(10^6)
GHZ,GZ: GHz(10^9)
■ Example SS△1MHZ
■ Restrictions according to modeltype and options.
If equipment is MS2665C, upper limit of f is equal to 21.2 GHz.
If equipment is MS2667C, upper limit of f is equal to 30.0 GHz.

SSS

SSS Scroll Step Size

■ Function Sets the scroll step size.

Header	Program command	Query	Response
SSS	SSS△n	SSS?	SSS△n

■ Value of n
1: 1div
2: 2div
5: 5div
1Ø: 10div

■ Suffix code None

■ Initial setting 2: 2div

■ Example SSS△1

ST

ST Sweep Time

■ Function Sets the frequency sweep time/time span.

Header	Program command	Query	Response
ST	ST△t ST△a	ST?	t t=12.5 to 1000000000 Transfers the data with no suffix code in units of 1 µs.

■ Value of t 12.5 µs to 1000 s (20 ms to 1000 s for frequency axis)

■ Value of a
UP: SWT UP
DN: SWT DOWN
AUTO: SWT AUTO

■ Suffix code
t: None: ms
US: µs
MS: ms
S: s
a: None

■ Initial setting Calculated value when AUTO is selected for SWT

■ Example
ST△AUTO
ST△2ØMS

■ Restrictions according to model type and options

If there is no opt.04 high-speed time domain, the value of t becomes 20 ms to 1000 s.

STF

STF Start Frequency

■ Function Sets the start frequency (same function as FA).

Header	Program command	Query	Response
STF	STF△f	STF?	STF△f f=-100000000 to 0 to 40000000000 Transfers the data with no suffix code in units of 1 Hz.

■ Value of f –100MHz to 40.0 GHz

■ Suffix code
None: Hz(10^0)
HZ: Hz(10^0)
KHZ,KZ: kHz(10^3)
MHZ,MZ: MHz(10^6)
GHZ,GZ: GHz(10^9)

■ Initial setting f=0Hz

■ Example
STF△123MHZ
STF△45.6KHZ

■ Restrictions according to model type and options
If equipment is MS2665C, upper limit of f is equal to 21.2 GHz.
If equipment is MS2667C, upper limit of f is equal to 30.0 GHz.

STPB

STPB Stop bit

■ Function Specifies the RS232C stop bit.

Header	Program command	Query	Response
STPB	STPB△n	STPB?	STPB△n

■ Value of n
1: 1 bit
2: 2 bit

■ Suffix code None

■ Initial setting 1: 1 bit

■ Example STPB△2

SV

SV Save Data into Internal Registor

■ Function Saves trace data/parameter data to the built-in register (same function as RGSV).

Header	Program command	Query	Response
SV	SV△n	————	————

■ Value of n 1 to 12 (Memory No.)
■ Suffix code None
■ Example SV△1

SVBMP

SVBMP Save BMP format file

■ Function Saves screen data(dot) at memory card using BMP format.

Header	Program command	Query	Response
SVBMP	SVBMP SVBMP△n	————	————

■ Value of n 1 to 999 (File No.) When omitted, number is appended automaticallay.
■ Suffix code None
■ Example SVBMP△1

SVM

SVM Save Data into Memory Card

■ Function

Saves the measurement conditions (parameters) and measured results (traces) to memory card.

Header	Program command	Query	Response
SVM	SVM△n	————	————

■ Value of n 1 to 99 (File No.)
■ Suffix code None
■ Example SVM△17
SVM△2

SWP

SWP Single Sweep/ Sweep Status

■ Function

Executes single sweep/Responds to sweep status (sweep completed/sweep in progress).
When accepted by the spectrum analyzer, the SWP command causes a single sweep to be executed by setting the sweep mode to 'SINGLE'.
The next command waits without being processed until its single sweep is completed (same function as TS). The SWP? Query command is used to Query the current sweep status (sweep completed/sweep in progress).

Header	Program command	Query	Response
SWP	SWP	SWP?	SWP△sw

■ Value of sw Ø: Sweep completed
1: Sweep progress

■ Example SWP
SWP?

SWSTART

SWSTART Restart Sweep

■ Function Restarts the sweep.

Header	Program command	Query	Response
SWSTART	SWSTART	————	————

■ Example `SWSTART`

SWSTOP

SWSTOP Stop Sweep

■ Function Stops the sweep.

Header	Program command	Query	Response
SWSTOP	SWSTOP	————	————

■ Example `SWSTOP`

SWT

SWT Sweep Time

■ Function Sets the frequency sweep time/time span (same function as ST).

Header	Program command	Query	Response
SWT	SWT△t	SWT?	SWT△t t=12.5 to 1000000000 Transfers the data with no suffix code in units of 1 μs.

■ Value of t 12.5 μs to 1000 s (20 ms to 1000 s for frequency domain)

■ Suffix code
None: ms
US: μs
MS: ms
S: s

■ Initial setting Calculated value when AUTO is selected for SWT

■ Example
SWT△1S
SWT△2ØMS

■ Restrictions according to model type and options
If there is no opt.04 high-speed time domain, the t becomes 20 ms to 1000 s.

TDLY

TDLY Delay Time

■ Function Sets the delay time from the point where trace time triggering occurs.

Header	Program command	Query	Response
TDLY	TDLY△t	TDLY?	t t=–1000000000 to 65500 Transfers the data with no suffix code in units of 1 µs.

■ Value of t –1000s to 65.5ms
■ Suffix code None: ms
US: µs
MS: ms
S: s
■ Initial setting Ø: 0s
■ Example TDLY △2ØMS
■ Restrictions according to model type and options
If there is no opt.06 Trigger/gate circuit, this command is invalid.

TEMP

TEMP Select Template

■ Function Selects one of the function templates.

Header	Program command	Query	Response
TEMP	TEMP△n	TEMP?	n

■ Value of n 1 to 5 (Template No.)
■ Suffix code None
■ Initial setting 1
■ Example TEMP△1

TEMPLOAD

TEMPLOAD Load Template data

■ Function Reads out template data from an external file.

Header	Program command	Query	Response
TEMPLOAD	TEMPLOAD△n	————	————

■ Value of n 1 to 99
■ Suffix code None
■ Example TEMPLOAD△1

TEMPMCL

TEMPMCL Cancel Moving Value

■ Function Returns a template movement to 0.

Header	Program command	Query	Response
TEMPMCL	TEMPMCL	————	————

■ Example TEMPMCL

TEMPMSV

TEMPMSV Save Moved Template Data

■ Function Stores the moved template data in the original template area.

Header	Program command	Query	Response
TEMPMSV	TEMPMSV	————	————

■ Example TEMPMSV

TEMPMVX

TEMPMVX Template Move X

■ Function Moves the template line along the X axis.

Header	Program command	Query	Response
TEMPMVX	TEMPMVX△t t=–1000 to 1000s	TEMPMVX?	t

■ Value of t –1000 to 1000s
■ Suffix code None: ms
US: μs
MS: ms
S: s
■ Initial setting Ø: 0s
■ Example TEMPMVX△1ØMS

TEMPMVY

TEMPMVY Template Move Y

■ Function Moves the template line along the Y axis.

Header	Program command	Query	Response
TEMPMVY	TEMPMVY△l	TEMPMVY?	l

■ Value of l –200.00dB to 200.00dB
■ Suffix code None: dB
DB,DBM,DM:dB
■ Initial setting Ø: 0dB
■ Example TEMPMVY△-2.5dB

TEMPSAVE

TEMPSAVE Save Template data

■ Function Moves the internal template data to an external file.

Header	Program command	Query	Response
TEMPSAVE	TEMPSAVE△n	————	————

■ Value of n 1 to 99
■ Suffix code None
■ Example TEMPSAVE△1

TEMPSLCT

TEMPSLCT Template Limit Line Select

■ Function Selects the Limit Line used for evaluating the measured results using the template functions.

Header	Program command	Query	Response
TEMPSLCT	TEMPSLCT△a,sw	TEMPSLCT?△a	sw sw=ON,OFF

■ Value of a
UP1: LIMIT1 UPPER
UP2: LIMIT2 UPPER
LW1: LIMIT1 LOWER
LW2: LIMIT2 LOWER
■ Value of sw
1,ON: ON
Ø,OFF: OFF
■ Suffix code None
■ Initial setting OFF
■ Example TEMPSLCT△UP1,ON

TEN

TEN Title Entry

■ Function Registers the title character string.

Header	Program command	Query	Response
TEN	TEN△x,y,text	———	———

■ Value of x,y X and Y values at display start point
(Do not use even if specified. Display location is fixed.)
■ Value of text Character string within 19 characters enclosed by double or single quotes.
■ Suffix code None
■ Example TEN△Ø,Ø,"TITLE SAMPLE"

TEXPAND

TEXPAND Time Expand

■ Function Turns ON/OFF the trace time-expansion functions.

Header	Program command	Query	Response
TEXPAND	TEXPAND△sw	TEXPAND?	sw sw=ON,OFF

■ Value of sw 1,ON: ON
Ø,OFF: OFF
■ Suffix code None
■ Example TEXPAND△ON

TIME

TIME Time

■ Function Sets the time of the built-in clock.

Header	Program command	Query	Response
TIME	TIME△hh,mm,ss	TIME?	hh,mm,ss

■ Value of hh 00 to 23 (Time)
■ Value of mm 00 to 59 (Minute)
■ Value of ss 00 to 59 (Second)
■ Suffix code None
■ Example TIME △Ø8,3Ø,ØØ

TIMEDSP

TIMEDSP Time Display

■ Function Sets time display on or off.

Header	Program command	Query	Response
TIMEDSP	TIMEDSP△sw	TIMEDSP?	sw

■ Value of sw ON: ON
OFF: OFF
■ Suffix code None
■ Initial setting OFF: Off
■ Example TIMEDSP△ON

TITLE

TITLE Title Entry

■ Function Registers the title character string (same function as KSE).

Header	Program command	Query	Response
TITLE	TITLE△text	TITLE?	text

■ Value of text Character string within 32 characters enclosed by single or double quotes.
■ Example TITLE△"MS2665"
TITLE△'SPECTRUM ANALYZER'

TLV

TLV Trigger Level

■ Function Sets the threshold level of sweep the start trigger when the trigger source is video and Ext mode.

Header	Program command	Query	Response
TLV	TLV△l	TLV?	TLV△l

■ Value of ℓ
For EXT: –10.0 to +10.0 (0.1 VStep)
For video and log: –100 to 0 (1dBStep)
For video and linear: 0 to 100 (1%Step)
For video and FM: –100 to 100 (2%Step)
For video (wide): HIGH,MID,LOW

■ Suffix code
When the trigger source is video and the step is log
None: dB
DB: dB
When the trigger source is EXT
None: V
V: V
In other case
None

■ Initial setting Ø

■ Example TLV△-5Ø

■ Restrictions according to model type and options
If there is no opt.06 trigger/gate circuit, this command is invalid.

TM

TM Trigger

■ Function Sets the trigger switch and trigger source (same function as TRG).

Header	Program command	Query	Response
TM	TM△a	TM?	a

■ Value of a
FREE: FREERUN
VID: VIDEO
WIDEVID: wide IF Video
LINE: LINE
EXT: EXT

■ Suffix code None

■ Initial setting FREE: FREERUN

■ Example TM△FREE

■ Restrictions according to model type and options

If there is no opt.06 trigger/gate circuit, this command is invalid.

TMCNT?

TMCNT? Time Count Read

■ Function Reads the values counted by the integrating meter which integrates the time or which electricity has been turned on.

Header	Program command	Query	Response
TMCNT?	————	TMCNT?	t t = Transfers the data with no suffix code in units of 1 hr.

■ Example TMCNT?

TMMD

TMMD Trace Time Storage Mode

■ Function Selects the mode for processing the trace TIME waveform.

Header	Program command	Query	Response
TMMD	TMMD△n	TMMD?	TMMD△n

■ Value of n
Ø: NORMAL
1: MAX HOLD
2: AVERAGE
3: MIN HOLD
4: CUMULATIVE
5: OVER WRITE

■ Suffix code None

■ Initial setting Ø: NORMAL

■ Example TMMD△Ø

TMWR

TMWR Trace Time Write Switch

■ Function Controls writing of the waveform to trace TIME.

Header	Program command	Query	Response
TMWR	TMWR△sw	TMWR?	TMWR△sw sw=ON,OFF

■ Value of sw `1,ON:` ON
`Ø,OFF:` OFF
■ Suffix code None
■ Initial setting `ON:` ON
■ Example `TMWR△ON`

TOUT

TOUT RS232C Time Out

■ Function Sets the time-out time for the RS232C WRITE function.

Header	Program command	Query	Response
TOUT	TOUT△t	TOUT?	t

■ Value of t `Ø:` Infinite (wait infinitely)
`1 to 255:` 1 to 255s(every 1 s step)
■ Suffix code None
■ Initial setting `3Ø:` 30s
■ Example `TOUT△1Ø`

TRG

TRG Trigger

■ Function Sets the trigger switch and trigger source (same function as TM).

Header	Program command	Query	Response
TRG	TRG△n	TRG?	TRG△a

■ Value of n
- Ø: FREERUN
- 1: VIDEO
- 2: LINE
- 3: EXT
- 7: WIDE IF VIDEO

■ Suffix code None

■ Initial setting Ø: FREERUN

■ Example TRG△Ø

■ Restrictions according to model type and options

If there is no opt.06 trigger/gate circuit is used, this command is invalid.

TRGLVL

TRGLVL Trigger Level

■ Function
Sets the sweep-start trigger level when the trigger source = VIDEO, WIDE IF VIDEO,EXT ±10V.

Header	Program command	Query	Response
TRGLVL	TRGLVL△l	TRGLVL?	l

■ Value of l

–10.0 to +10.0 (0.1 Step) :	when the trigger source is EXT (±10V)(V units)
–100 to +100(1 Step) :	when the trigger source is VIDEO and the scale is LOG (dB units)
0 to 100 (1 step):	When the trigger source is VIDEO and the scale is LIN (% units)
–100 to +100 (2 step):	When the trigger source is VIDEO and FM monitor (% units)

■ Suffix code
When the trigger source is VIDEO and the scale is LOG

None:	dB
DB:	dB

When the trigger source is EXT

None:	V
V:	V

In other case

None

■ Initial setting
l= -40

■ Example
```
TRGLVL△-1Ø.Ø
TRGLVL△9.9
```

■ Restrictions according to model type and options

If there is no opt.06 trigger/gate circuit is used, this command is invalid.

TRGS

TRGS Trigger Switch

■ Function Switches the trigger switch to Free run or Triggered.

Header	Program command	Query	Response
TRGS	TRGS△a	TRGS?	a

■ Value of a FREE: FREERUN
TRGD: TRIGGERED
■ Suffix code None
■ Initial setting FREE: FREERUN
■ Example TRGS△FREE
■ Restrictions according to model type and options
If there is no opt.06 trigger/gate circuit, this command is invalid.

TRGSLP

TRGSLP Trigger Slope

■ Function Selects the rising or falling slope of the trigger when trigger source is VIDEO or EXT mode.

Header	Program command	Query	Response
TRGSLP	TRGSLP△a	TRGSLP?	a

■ Value of a RISE: Rising edge
FALL: Falling edge
■ Suffix code None
■ Initial setting RISE: Rising edge
■ Example TRGSLP△RISE
■ Restrictions according to model type and options
If there is no opt.06 trigger/gate circuit, this command is invalid.

TRGSOURCE

TRGSOURCE Trigger Source

■ Function Selects the trigger source. The trigger switch setting is not changed by this command.

Header	Program command	Query	Response
TRGSOURCE	TRGSOURCE△a	TRGSOURCE?	a

■ Value of a
VID: VIDEO
WIDEVID: WIDE IF VIDEO
LINE: LINE
EXT: EXT

■ Suffix code None

■ Initial setting VID: VIDEO

■ Example TRGSOURCE△VID

■ Restrictions according to model type and options

If there is no opt.06 trigger/gate circuit, this command is invalid.

TRM

TRM Terminator

■ Function Sets the terminator of the Response data transferred on the GPIB.

Header	Program command	Query	Response
TRM	TRM△n	———	———

■ Value of n
Ø: LF
1: CR/LF

■ Suffix code None

■ Initial setting Ø: LF(provided the terminator already registered is not initialized)

■ Example
TRM△Ø
TRM△1

TS

TS Take Sweep

■ Function Executes a single sweep synchronously (same function as SWP).

Header	Program command	Query	Response
TS	TS	————	————

■ Example TS

TSAVG

TSAVG Take Sweep with Averaging

■ Function Performs synchronous sweeping the number of times specified in the current Averaging setting.

Header	Program command	Query	Response
TSAVG	TSAVG	————	————

■ Example TSAVG

TSHOLD

TSHOLD Take Sweep with Max/Min Holding

■ Function Performs synchronous sweeping by the number of times specified in the current holding setting.

Header	Program command	Query	Response
TSHOLD	TSHOLD	———	———

■ Example TSHOLD

TSL

TSL Trigger Slope

■ Function Selects triggering on the rising or falling trigger slope.

Header	Program command	Query	Response
TSL	TSL△sw	TSL?	TSL△sw

■ Value of sw Ø: Fall
1: Rise
■ Suffix code None
■ Initial setting 1: Rise
■ Example TSL△Ø
■ Restrictions according to model type and options
If there is no opt.06 trigger/gate circuit, this command is invalid.

TSP

TSP Time Span

■ Function Sets the time span of the trace.

Header	Program command	Query	Response
TSP	TSP△t	TSP?	t t=12.5 to 1000000000 Transfers the data with no suffix code in units of 1 µs

■ Value of t 12.5µs to 1000s
■ Suffix code None: ms
US: µs
MS: ms
S: s
■ Initial setting 200ms
■ Example TSP△1ØØ
TSP△1ØØS
■ Restrictions according to model type and options
If there is no opt.04 high-speed time domain, the value of t becomes 20 ms to 1000 s.

TTL

TTL Title Display Switch

■ Function Switches the title display to ON/OFF.

Header	Program command	Query	Response
TTL	TTL△sw	TTL?	TTL△sw sw=ON,OFF

■ Value of sw 1,ON: ON
Ø,OFF: OFF
■ Suffix code None
■ Initial setting OFF: OFF
■ Example TTL△ON

TZONE

TZONE Expand Zone

■ Function Switches the time expansion (magnified display) ON/OFF.

Header	Program command	Query	Response
TZONE	TZONE△sw	TZONE?	sw sw=ON,OFF

■ Value of sw 1,ON: ON
Ø,OFF: OFF
■ Suffix code None
■ Initial setting OFF: OFF
■ Example TZONE△ON
■ Restrictions according to model type and options
If there is no opt.06 trigger/gate circuit, this command is invalid.

TZSP

TZSP Expand Zone Span

■ Function Sets the zone for time expansion (magnified display).

Header	Program command	Query	Response
TZSP	TZSP△t	TZSP?	t t=12.5 to 1000000000 Transfers the data with no suffix code in units of 1 µs

■ Value of t 12.5µs to 1000s
■ Suffix code None: ms
US: µs
MS: ms
S: s
■ Initial setting 200ms
■ Example TZSP△1ØMS
■ Restrictions according to model type and options
If there is no opt.06 trigger/gate circuit, this command is invalid.

TZSPP

TZSPP Expand Zone Span point

■ Function Specifies the width of the Expand Zone in term of the number of points.

Header	Program command	Query	Response
TZSPP	TZSPP△p	TZSPP?	p

■ Value of p 1 to 500
■ Suffix code None
■ Initial setting 1ØØ: 101 points (2 div)
■ Example TZSPP△51
■ Restrictions according to model type and options
If there is no opt.06 trigger/gate circuit, this command is invalid.

TZSTART

TZSTART Expand Zone Start

■ Function Sets the start time for time expansion (magnified display).

Header	Program command	Query	Response
TZSTART	TZSTART△t	TZSTART?	t t=–1000000000 to 65500 Transfers the data with no suffix code in units of 1 µs

■ Value of t –1000s to 65.5ms
■ Suffix code None: ms
US: µs
MS: ms
S: s
■ Initial setting 0s
■ Example TZSTART△1ØMS
■ Restrictions according to model type and options
If there is no opt.06 trigger/gate circuit, this command is invalid.

TZSTARTP

TZSTARTP Expand Zone Start point

■ Function Specifies the start point of the Expand Zone in terms of the number of point.

Header	Program command	Query	Response
TZSTARTP	TZSTARTP△p	TZSTARTP?	p

■ Value of p 0 to 500
■ Suffix code None
■ Initial setting 2ØØ: 200 point
■ Example TZSTARTP△1ØØ
■ Restrictions according to model type and options
If there is no opt.06 trigger/gate circuit, this command is invalid.

UCL?

UCL? Query Uncal Status

■ Function Reads out the UNCAL status.

Header	Program command	Query	Response
UCL?	————	UCL?	UCL△n

■ Value of n
Ø: NORMAL
1: During UNCAL

■ Example UCL?

UNC

UNC Uncal Display ON/OFF

■ Function Specifies whether 'UNCAL' is displayed when UNCAL occurs.

Header	Program command	Query	Response
UNC	UNC△sw	UNC?	UNC△sw sw=ON,OFF

■ Value of sw
1,ON: ON
Ø,OFF: OFF

■ Suffix code None

■ Initial setting ON: ON

■ Example UNC△ON

UNLOCKCOUNT

UNLOCKCOUNT Unlock count for frequency domain sweep

■ Function Set the count of sweeps in one cycle for lock in frequency domain operation.

Header	Program command	Query	Response
UNLOCKCOUNT	UNLOCKCOUNT△n	UNLOCKCOUNT?	n

■ Value of n 1 to 100
■ Suffix code None
■ Initial setting 1Ø
■ Example UNLOCKCOUNT△20 Performs a frequency lock operation once in every 20 sweeps.

UNT

UNT Unit for Log Scale

■ Function Sets the display unit system in LOG scale mode.

Header	Program command	Query	Response
UNT	UNT△a	UNT?	UNT△a

■ Value of a
Ø: dBm
1: dBμV
2: dBmV
3: V
4: dBμV(emf)
5: W

■ Suffix code None
■ Initial setting Ø: dBm
■ Example UNT△Ø

VAR

VAR Write value to common variable

■ Function Write value to common variable used at PTA library.

Header	Program command	Query	Response
VAR	VAR△a,b	VAR?△a	b

■ Value of a Common variable name
(Integer/Real-number numeric variable name,alpha-numeric characters within 7 characters)VAVG

■ Value of b Value to be written (Integer or real-number)

■ Suffix code None

■ Example VAR△COOMAB,1Ø.5
VAR△XYZ%,1ØØ

VAVG

VAVG Average

■ Function Sets averaging ON or OFF and sets the number of averaging processes.

Header	Program command	Query	Response
VAVG	VAVG△sw VAVG△n	VAVG?	n

■ Value of sw 1,ON: ON
Ø,OFF: OFF

■ Value of n 2 to 1024: Number of averaging processes

■ Suffix code None

■ Initial setting 8: 8 times

■ Example VAVG△ON
VAVG△128

VB

VB Video Bandwidth

■ Function Sets the video bandwidth (same function as VBW).

Header	Program command	Query	Response
VB	VB△f VB△a	VB?	f f=1 to 3000000 or OFF Transfers the data with no suffix code in units of 1 Hz.

■ Value of f 1Hz to 3MHz
■ Value of a OFF: OFF
AUTO: AUTO
UP: VBW UP
DN: VBW DOWN
■ Suffix code f: None: Hz(10^0)
HZ: Hz(10^0)
KHZ,KZ: kHz(10^3)
MHZ,MZ: MHz(10^6)
GHZ,GZ: GHz(10^9)
a: None
■ Initial setting Calculated value when VBW=AUTO.
■ Example VB△3ØØHZ

VBCOUPLE

VBCOUPLE Couple Mode

■ Function Sets the coupled functions to commonly settable or independently settable at the frequency domain and time domain.

Header	Program command	Query	Response
VBCOUPLE	VBCOUPLE△a	VBCOUPLE?	a

■ Value of a COM: Common
IND: Independent
■ Suffix code None
■ Initial setting IND: Independent (the mode already registered is not initialized.)
■ Example VBCOUPLE△COM

VBR

VBR VBW/ RBW Ratio

■ Function: Sets the ratio of video bandwidth to resolution bandwidth when VBW is selected for AUTO.

Header	Program command	Query	Response
VBR	VBR△r	VBR?	r r=0.0001 to 100

■ Value of r: 0.0001 to 100 (1/3 sequence)
■ Suffix code: None
■ Initial setting: Trace A,B,BG:VBW/RBW RATIO=1
Trace TIME:VBW/RBW RATIO=1
■ Example: VBR△1

VBW

VBW Video Bandwidth

■ Function: Sets the video bandwidth.

Header	Program command	Query	Response
VBW	VBW△n	VBW?	VBW△n

■ Value of n:

Ø:	1Hz	8:	3Hz
1:	10Hz	9:	30Hz
2:	100Hz	1Ø:	300Hz
3:	1kHz	11:	3kHz
4:	10kHz	12:	30kHz
5:	100kHz	13:	300kHz
6	OFF	14:	3MHz
7:	1MHz		

■ Suffix code: None
■ Initial setting: Calculated value when VBW is selected for AUTO
■ Example: VBW△3

VIEW

VIEW View

■ Function Stops writing of the waveform data.

Header	Program command	Query	Response
VIEW	VIEW△tr	————	————

■ Value of tr

TRA: Trace A
TRB: Trace B
TRBG: Trace BG
TRTIME: Trace TIME

■ Suffix code None

■ Example VIEW△TRB

XCH

XCH Exchange Traces

■ Function Exchanges the specified wave data of traces.

Header	Program command	Query	Response
XCH	XCH△tr1,tr2	————	————

■ Value of tr1,tr2 TRA: Trace-A
TRB: Trace-B
■ Suffix code None
■ Example XCH△TRA,TRB

XMA

XMA Trace A Spectrum Data

■ Function Writes/reads the spectrum data to/from trace A (main trace) memory.

Header	Program command	Query	Response
XMA	XMA△p,b	XMA?△p,d	b1,b2,b3 ...... (ASCII) b1 b2 b3 ...... (BINARY)

■ Value of p 0 to 500(point No.)
■ Value of b LOG scale: Integer of 0.01 dBm unit (independent of display unit system)

LIN scale: $b = \dfrac{\text{Voltage value (V)}}{\text{reference level (V)}} \times 10000$

When binary format is specified for response data, data for each point is composed of two bytes. The high-order byte is sent first.
■ Value of d 1 to 501(number of points)
■ Example XMA△1,-2ØØØ
XMA?△1,2(reads two-point data items starting from point 1)

XMB

XMB Trace B Spectrum Data

■ Function Writes/reads the spectrum data to/from to trace B (main trace) memory.

Header	Program command	Query	Response
XMB	XMB△p,b	XMB?△p,d	b1,b2,b3 ...... (ASCII) b1 b2 b3 ...... (BINARY)

■ Value of p 0 to 500(point No.)

■ Value of b `LOG scale:` Integer of 0.01 dBm unit (independent of display unit system)

`LIN scale:` $b = \dfrac{\text{Voltage value (V)}}{\text{reference level (V)}} \times 10000$

When binary format is specified for response data, data for each point is composed of two bytes. The high-order byte is sent first.

■ Value of d 1 to 501(number of points)

■ Example `XMB△1,-2ØØØ`

`XMB?△1,2`(reads two-point data items starting from point 1)

XMG

XMG Trace BG Spectrum Data

■ Function Writes/reads the spectrum data to/from to trace BG memory.

Header	Program command	Query	Response
XMG	XMG△p,b	XMG?△p,d	b1,b2,b3 ...... (ASCII) b1 b2 b3 ...... (BINARY)

■ Value of p 0 to 500(point No.)

■ Value of b `LOG scale:` Integer of 0.01 dBm unit (independent of display unit system)

`LIN scale:` $b = \dfrac{\text{Voltage value (V)}}{\text{reference level (V)}} \times 10000$

When binary format is specified for response data, data for each point is composed of two bytes. The high-order byte is sent first.

■ Value of d 1 to 501(number of points)

■ Example `XMG△1,-2ØØØ`

`XMG?△1,2`(reads two-point data items from point 1)

XMT

XMT Trace TIME Spectrum Data

■ Function Write/reads the spectrum data to/from the trace TIME memory.

Header	Program command	Query	Response
XMB	XMT△p,b	XMT?△p,d	b1,b2,b3 ...... (ASCII) b1 b2 b3 ...... (BINARY)

■ Value of p 0 to 500(point No.)

■ Value of b `LOG scale:` Integer of 0.01 dBm unit (independent of display unit system)

`LIN scale:` $b = \dfrac{\text{Voltage value (V)}}{\text{reference level (V)}} \times 10000$

When binary format is specified for response data, data for each point is composed of two bytes. The high-order byte is sent first.

■ Value of d 1 to 501(number of points)

■ Example `XMT△1,-2ØØØ`

`XMT?△1,2`(reads two-point data items starting from point 1)

ZEROSPNMODE

ZEROSPNMODE Zero Span Sweep mode

■ Function Set the mode inside a spectrum analyzer for realizing zero span.

Header	Program command	Query	Response
ZEROSPNMODE	ZEROSPNMODE△a	ZEROSPNMODE?	a

■ Value of a DIGITAL: Digital mode

ANALOG: Analog mode

■ Suffix code None

■ Initial setting DIGITAL: Digital zero span

■ Example ZEROSPNMODE△ANALOG

■ Supplement This function is used when you want to use sweep signals, X-out and Z-out also in a zero span sweep. In this case, set to "ANALOG".
In a normal operation, use "DIGITAL" mode.

∗CLS

∗CLS Clear Status Command

■ Function Clears the status byte register.

Header	Program command	Query	Response
∗CLS	∗CLS	———	———

■ Example ∗CLS

∗ESE

∗ESE Standard Event Status Enable

■ Function Sets or clears the standard status enable register.

Header	Program command	Query	Response
∗ESE	∗ESE△n	∗ESE△?	n

■ Value of n 0 to 255
■ Example ∗ESE△2Ø
∗ESE?

*ESR?

*ESR? Standard Event Status Register Query

■ Function Returns the current value in the standard event status register.

Header	Program command	Query	Response
*ESR	————	*ESR?	n

■ Value of n 0 to 255

■ Example *ESR?

---

*IDN?

*IDN? Identification Query

■ Function Returns the manufacturer name, model number etc. of the equipment.

Header	Program command	Query	Response
*IDN	————	*IDN?	ANRITSU,id,ØØØØ,n

■ Value of id MS2665C (In case of MS2665C)
MS2667C (In case of MS2667C)
MS2668C (In case of MS2668C)

■ Value of n 1 to 99(firmware version No.)

■ Example *IDN?

*OPC

*OPC Operation Complete Command

■ Function Sets bit 0 in the standard event status register when all pending selected device operations have been completed.

Header	Program command	Query	Response
*OPC	*OPC	———	———

■ Example *OPC

---

*OPC?

*OPC? Operation Complete Query

■ Function Sets the output queue to 1 to generate a MAV summary message when all pending selected device operations have been completed.

Header	Program command	Query	Response
*OPC?	———	*OPC?	1

■ Example *OPC?

∗RST

∗RST Reset Command

■ Function Resets the device to the third level.

Header	Program command	Query	Response
∗RST	∗RST	———	———

■ Example ∗RST

∗SRE

∗SRE Service Request Enable Command

■ Function Sets the bits in the service request enable register.

Header	Program command	Query	Response
∗SRE	∗SRE△n	∗SRE?	n

■ Value of n 0 to 63, 128 to 191(current value of the service request enable register)
■ Example ∗SRE

∗STB?

∗STB? Read Status Byte Command

■ Function Returns the current values of the status bytes including the MSS bit.

Header	Program command	Query	Response
∗STB	————	∗STB?	n

■ Value of n

Bit	Bit weight	Bit name	Condition of status byte register
7	128	————	0= Not used
6	64	MSS	0= Service not requested 1=Service requested
5	32	ESB	0=Event status not generated 1= Event status generated
4	16	MAV	0=No data in output queue 1= Data in output queue
3	8	————	0= Not used
2	4	ESB(END)	0= Event status not generated 1= Event status generated
1	2	————	0= Not used
0	1	————	0= Not used

■ Example ∗STB?

*TRG

*TRG Trigger Command

■ Function

Same function as that of IEEE488 GET-group-execute-trigger bus command. For this command, the MS2665C/67C/68C executes a single sweep (same function as SWP.)

Header	Program command	Query	Response
*TRG	*TRG	————	————

■ Example

*TRG

*TST

*TST Self Test Query

■ Function

Executes an internal self-test and returns the details of any errors.

Header	Program command	Query	Response
*TST	————	*TST?	n

■ Value of n

Ø: Self-test completed with no errors.
-32767 to -1,
1 to 327671: Self-test was not completed, or was completed but with errors.

■ Example

*TST?

*WAI

*WAI Wait-to-Continue Command

■ Function Keeps the next command on stand-by while the device is executing a command.

Header	Program command	Query	Response
*WAI	*WAI	————	————

■ Example *WAI

library name

library name Execute PTA Library

■ Function Executes PTA library.

Header	Program command	Query	Response
LIBRARY NAME	LIBRARY NAME	————	————

■ Value of library name

PTA library name (alpha-numeric characters within 8 characters)

VAR△XYZ%,1ØØ

APPENDIXES

TABLE OF CONTENTS

APPENDIX A
TABLE OF MS2665C/67C/68C DEVICE-DEPENDENT INITIAL SETTINGS

Table A Device-Dependent Initial Settings (1/5)

Group	Outline	Control item	Initial setting data		
			TRACE-A,B	TRACE-TIME	TRACE-BG
Frequency	Selects the mode for setting a frequency band.	FREQUENCY MODE	START-STOP		
	Sets the start frequency	START FREQUENCY	0 Hz	---------	0Hz
	Sets the center frequency	CENTER FREQUENCY	(*1)		
	Sets the stop frequency	STOP FREQUENCY	(*2)	----------	(*2)
	Sets the frequency span	FREQUENCY SPAN	(*2)	∗0 Hz	(*2)
	Sets the center-frequency step size	CENTER FREQ STEP SIZE	1 GHz		
	Sets the scroll step size	SCROLL STEP SIZE	2 div		
	Select Band	BAND SELECT	AUTO		
Level	Sets the reference level	REFERENCE LEVEL	–10 dBm		
	Set the reference level step size	REF LEVEL STEP SOZE	AUTO:1div		
	Sets the scale mode	SCALE MODE	LOG	LOG	∗LOG
	Sets the LOG scale	LOG SCALE	10 dB/div	10 dB/div	∗10 dB/div
	Sets the LIN scale	LIN SCALE	10%/div	10%/div	---------
	Sets the LOG unit system	LOG SCALE UNIT	Not initialized ∗RST: dBm		
	Sets the reference level offset	REF LEVEL OFFSET	OFF		
	Sets the reference level offset value	OFFSET VALUE	0 dBm		
	Sets the display line	DISPLAY LINE	OFF		
	Sets the display line level	DISPLAY LINE LEVEL	–60 dBm		
	Selects the ABS or RELmarker level	MARKER LEVEL ABS/REL	A:ABS B:ABS	ABS	ABS
	Sets the correction factor	CORRECTION	Not initialized ∗RST: OFF		
	Sets the correction factor number	CORRECTION FACTOR No.	∗RST: 1		
	Sets the input impedance	INPUT INPEDANCE	50Ω		

(*1) For the MS2665C; 10.6 GHz, for the MS2667C; 15.0 GHz, for the MS2668C; 20.0 GHz

(*2) For the MS2665C; 21.2 GHz, for the MS2667C; 30.0 GHz, for the MS2668C; 40.0 GHz

Table A Device-Dependent Initial Settings (2/5)

Group	Outline	Control item	Initial setting data		
			TRACE-A,B	TRACE-TIME	TRACE-BG
Display mode	Selects the display mode	DISPLAY MODE	TRACE-A		
	Selects the display format for TRACE-A/B	DISPLAY FORMAT (TRACE-A/B)	A<B		
	Selects the display format for TRACE-A/BG	DISPLAY FORMAT (TRACE-A/BG)	A<BG		
	Selects the display format for TRACE-A/TIME	DISPLAY FORMAT (TRACE-A/TIME)	A<TIME		
	Selects the mode for processing a waveform	TRACE STORAGE MODE	NORMAL	NORMAL	∗NORMAL
	Number of traces averaged	AVERAGE No.	8 times		
	Sets the separation of average sweep stops	AVERAGE SWEEP MODE	ON(PAUSE)		
	Sets the separation of hold sweep stops	HOLD SWEEP MODE	OFF(CONTINUOUS)		
	Selects the detection mode	DETECTION MODE	PEAK	SAMPLE	∗PEAK
	Sets the delay time	DELAY TIME	------	0 s	------
	Sets the time span	TIME SPAN	------	# 200 ms	------
	Sets the time expansion zone to ON/OFF	EXPAND ZONE ON/OFF	------	OFF	------
	Sets the expand mode to ON/OFF	EXPAND ON/OFF	------	OFF	------
	Sets the FM monitor to ON/OFF	FM MONITOR	------	OFF	------
	Sets the bandwidth for demodulating FM	FM RANGE	------	200 kHz/div	------
	Switches the coupling to AC/DC to monitor FM waveforms	FM COUPLING	------	AC COUPLING	------
	Sets the active marker when display mode is trace A/B	TRACE-A/B ACTIVE MKR	TRACE-A	------	------
	Selects the marker mode	MARKER MODE	NORMAL		
	Specifies the zone-marker center	ZONE MAKER CENTER	250 point	250 point	250 point
	Specifies the zone-marker width	ZONE MAKER WIDTH	51 point(1 div)	∗1 point	501 point
	Marker search mode	MAKER SEARCH MODE	PEAK		
	Sets the multi marker mode to ON/OFF	MULTI MARKER MODE	OFF		
	Sets the multi marker list to ON/OFF	MULTI MARKER LIST	OFF		
	Multi marker list frequency AES/REL	MULTI MARKER LOST FREQ	ABS		
	Multi marker list level ABS/REL	MULTI MARKER LOST LEVEL	ABS		
	Sets the 'n'th multi marker to ON/OFF (No.1 to 10)ON/OFF	MULTI MARKER ON/OFF	Not initialized RST: No.1 = ON, No.2 to 10 = OFF		
	Selects the active multi marker	ACTIVE MARKER No.	Not initialized ∗RST: No.1		
	Search resolution	SEARCH RESOLUTION	10 dB		
	Search threshold	THRESHOLD	OFF		

Table A Device-Dependent Initial Settings (3/5)

Group	Outline	Control item	Initial setting data		
			TRACE-A,B	TRACE-TIME	TRACE-BG
Trace operation	A-B→A	A-B→A	OFF		
	A-B REFERENCE LINE	REFERENCE LINE	MIDDLE		
	Normalize(A - B None)	NORMALIZE	OFF		
Sweep function	Sets the sweep mode	SWEEP MODE	CONTINUOUS		
	Sets the zone sweep to ON/OFF	ZONE SWEEP	OFF	----------	
	Sets the tracking function to ON/OFF	TRACKING SWEEP	OFF	----------	
	Sets the gate sweep function to ON/OFF	GATE SWEEP	OFF		---------
	Sets the gate delay time	GATE DELAY	0 s		---------
	Sets the gate length	GATE LENGTH	1 ms		---------
	Sets the gate interval termination, internally or externally	GATE END	INTERNAL		---------
	Sets the trigger switch mode	TRIGGER SWITCH	FREE RUN	FREE RUN	*FREE RUN
	Sets the trigger source	TRIGGER SOURCE	VIDEO		---------
	Sets the external trigger level type	TRIGGER SOURCE(EXT)	INPUT1		---------
	Selects the trigger slope	TRIGGER SLOPE	RISE		---------
	Sets the trigger level	TRIGGER LEVEL	-40dB		---------
	Trigger level (WIDE IF VEDEO)	TRIGGER LEVEL (WIDE IF VIDEO)	HIGH		
Waveform writing/reading	Sets the trace write switch to ON/OFF	TRACE WRITE SWITCH	ON	ON	ON
	Sets the trace read switch to ON/OFF	TRACE READ SWITCH	ON	ON	ON
Coupled function	Selects the mode for setting the resolution bandwidth	RESOLUTION BANDWIDTH	AUTO	AUTO	*AUTO
	Selects the mode for setting the video bandwidth	VIDEO BAND WIDTH	AUTO	AUTO	*AUTO
	Selects the mode for setting the sweep time	SWEEP TIME	AUTO	AUTO	*AUTO
	Selects the mode for setting the RF attenuator	RF ATTENUATOR	AUTO		
	VBW/RBW ratio at VBW = AUTO	VBW/RBW RATIO	1	1	1
	RBW/Span ratio at RBW = AUTO	RBW/SPAN RATIO	0.01	0.01	0.01
	Sets the coupled functions to COMMON or INDEPENDENT between the frequency or time domain	COUPLE MODE (COMMON/INDEPENDENT)	Not initialized. When shipped from the factory: INDEPENDENT		
SAVE/ RECALL	Selects data to be recalled	RECALLED DATA	Not initialized. When shipped from the factory: View		
Hard copy/plot	Select the printer device mode	PRINTER MODE	Not initialized. When shipped from the factory: VP600		
	Print magnification	PRINT MAGNIFICATION	1x1		

Table A Device-Dependent Initial Settings (4/5)

Group	Outline	Control item	Initial setting data		
			TRACE-A,B	TRACE-TIME	TRACE-BG
Hard copy/plot	Sets the printer GPIB address	PRINTER GPIB ADDRESS	Not initialized. When shipped from the factory: 17		
	Selects the paper size for the plotter	PLOTTER PAPER SIZE	Not initialized. When shipped from the factory: A4		
	Selects the plotter output size	PLOTTER SIZE	Not initialized. When shipped from the factory: FULL		
	Selects the plot item	PLOT ITEM	Not initialized. When shipped from the factory: ALL		
Sound monitor	Selects the mode for monitoring the sound	AM/FM MONITOR	OFF		
	Adjusts the volume of the sound monitor	MONITOR VOLUME	10		
Measure function	Selects the item to be measured	MEASURE ITEM	OFF		
	Sets the counter to the specified resolution	COUNT RESOLUTION	1 kHz		
	Selects the occupied frequency bandwidth measurement method	OBW MEASURE METHOD	Not initialized *RST: N%		
	Sets the occupied frequency bandwidth to N%	OBW N% VALUE	Not initialized *RST: 99%		
	Sets the occupied frequency to X dB	OBW XdB VALUE	Not initialized *RST: 25dB		
	Selects the adjacent channel leakage power measurement method	ADJ-CH MEASURE METHOD	Not initialized *RST: R:TOTAL POWER		
	Selects the adjacent channel leakage power measurement method	ADJ-CH GRAPH	Not initialized *RST: ON		
	Selects the adjacent channel	ADJACENT CH SELECT	Not initialized *RST: BOTH SIDES		
	Sets adjacent separation 1	ADJACENT CH SEPARATION1	Not initialized *RST: 12.5 kHz		
	Sets the adjacent separation 2	ADJACENT CH SEPARATION2	Not initialized *RST: 25.0 kHz		
	Sets the adjacent channel bandwidth	ADJACENT CH BANDWIDTH	Not initialized *RST: 8.5 kHz		
	Sets the adjacent channel center line display	ADJ-CH CENTER LINE	Not initialized *RST: ON		
	Sets the adjacent channel band line display	ADJ-CH BAND LINE	Not initialized *RST: OFF		
	Selects the template	SELECT TEMPLATE	Not initialized *RST: No.1		
	Selects the template level	TEMPLATE LEVEL	Not initialized *RST: ABSOLUTE		
	Sets the template management function	MANEGE TEMPLATE	Not initialized		
	Selects the noise measurement method	NOISE MEASURE METHOD	Not initialized *RST: ABS		

Table A Device-Dependent Initial Settings (5/5)

Group	Outline	Control item	Initial setting data		
			TRACE-A,B	TRACE-TIME	TRACE-BG
Measure function	BURST POWER START POINT	BURST POWER MEASURE START POINT	100 point		
	BURST POWER STOP POINT	BURST POWER MEASURE STOP POINT	400 point		
Calibration	Frequency calibration	FREQ CAL	ON		
RS-232C	Band rate	BAUD RATE	2400		
	Parity	PARITY	OFF		
	Data bit	DATA BIT	8 bit		
	Stop bit	STOP BIT	1 bit		
	Time-out	TIME OUT	30 s		
GPIB	Sets the GPIB 2 self address	GPIB SELF ADDRESS	Not initialized. When shipped from the factory: 0		
	GPIB timeout time (including trigger sweep time out)	GPIB TIME OUT (TRIGGER SWEEP TIME OUT)			
	Sets the DSU (MC8104A) address	DATA STORAGE UNIT ADDRESS	Not initialized. When shipped from the factory: 19		
Title	Sets the title output to ON/OFF	TITLE ON/OFF	Not initialized. When shipped from the factory: ON		
	Selects the title data	TITLE DATA	Not initialized. When shipped from the factory: ALLSPACE		
CAL/ UNCAL	Displays couple failure	UNCAL DISPLAY	Not initialized. Initialized to ON at power-on.		
Spectrum data/ PMC/ETC	Sets the response data to ASCII/BINARY	RESPONSE DATA	Not initialized. When shipped from the factory: ASCII		
	Selects the media (PMC/floppy disk)	SLOT	Not initialized. When shipped from the factory: SLOT 1 (top)		
	Selects the terminator for LF/CR + LF	TERMINATOR	Not initialized. When shipped from the factory: LF		
Others	Power input status	POWER ON STATE	BEFORE POWER OFF		
	Parameter display system	PARAMETER DISPLAY TYPE	TYPE-1		
	Time display	TIME DISPLAY	OFF		
	Date display system	DATE DISPLAY MODE	YYYY/MM/DD		
	Comment column display system	COMMENT DISPLAY	OFF		
	Display color pattern	COLOR PATTERN	COLOR1		
	LCD display	LCD DISPLAY	ON		
	Composite mode	COMPOSITE MODE	NORMAL		

Note: • In the above table, in place of the parameters not initialized by the INIT command or P+reset key, the initial settings (indicated by *RST) initialized by the *RST command are listed. In place of the parameters not initialized by the *RST command, the values at the shipment are listed.
• An initial value marked with '*' is a fixed value.
• An initial value marked with '#' is the value at COUPLE MODE = COMMON.

APPENDIX B

ASCII*CODE TABLE

B7 B6 B5 →				0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
BITS				CONTROL		NUMBERS SYMBOLS		UPPER CASE		LOWER CASE	
B4	B3	B2	B1								
0	0	0	0	0 NUL 0 0	20 DLE 10 16	40 SP 20 32	60 0 30 48	100 @ 40 64	120 P 50 80	140 ˋ 60 96	160 p 70 112
0	0	0	1	1 GTL SOH 1 1	21 LLO DC1 11 17	41 ! 21 33	61 1 31 49	101 A 41 65	121 Q 51 81	141 a 61 97	161 q 71 113
0	0	1	0	2 NUL 2 2	22 DC2 12 18	42 " 22 34	62 2 32 50	102 B 42 66	122 R 52 82	142 b 62 98	162 r 72 114
0	0	1	1	3 ETX 3 3	23 DC3 13 19	43 # 23 35	63 3 33 51	103 C 43 67	123 S 53 83	143 c 63 99	163 s 73 115
0	1	0	0	4 SDC EOT 4 4	24 DCL DC4 14 20	44 S 24 36	64 4 34 52	104 D 44 68	124 T 54 84	144 d 64 100	164 t 74 116
0	1	0	1	5 PPC ENO 5 5	25 PPU NAK 15 21	45 % 25 37	65 5 35 53	105 E 45 69	125 U 55 85	145 e 65 101	165 u 75 117
0	1	1	0	6 ACK 6 6	26 SYN 16 22	46 & 26 38	66 6 36 54	106 F 46 70	126 V 56 86	146 f 66 102	166 v 76 118
0	1	1	1	7 BEL 7 7	27 ETB 17 23	47 ' 27 39	67 7 37 55	107 G 47 71	127 W 57 87	147 g 67 103	167 w 77 119
1	0	0	0	10 GET BS 8 8	30 SPE CAN 18 24	50 (28 40	70 8 38 56	110 H 48 72	130 X 58 88	150 h 68 104	170 x 78 120
1	0	0	1	11 TCT HT 9 9	31 SPD EM 19 25	51) 29 41	71 9 39 57	111 I 49 73	131 Y 59 89	151 i 69 105	171 y 79 121
1	0	1	0	12 LF A 10	32 SUB 1A 26	52 * 2A 42	72 : 3A 58	112 J 4A 74	132 Z 5A 90	152 j 6A 106	172 z 7A 122
1	0	1	1	13 VT B 11	33 ESC 1B 27	53 ÷ 2B 43	73 ; 3B 59	113 K 4B 75	133 [5B 91	153 k 6B 107	173 { 7B 123
1	1	0	0	14 FF C 12	34 FS 1C 28	54 , 2C 44	74 < 3C 60	114 L 4C 76	134 \ 5C 92	154 l 6C 108	174 ¦ 7C 124
1	1	0	1	15 CR D 13	35 GS 1D 29	55 - 2D 45	75 = 3D 61	115 M 4D 77	135 [5D 93	155 m 6D 109	175 } 7D 125
1	1	1	0	16 SO E 14	36 RS 1E 30	56 . 2E 46	76 > 3E 62	116 N 4E 78	136 ∧ 5E 94	156 n 6E 110	176 ~ 7E 126
1	1	1	1	17 SI F 15	37 US 1F 31	57 / 2F 47	77 ? 3F 63	117 O 4F 79	137 UNT — 5F 95	157 o 6F 111	177 RUBOUT (DEL) 7F 127
				Address command	Universal command	Listen address		Talk address		Secondary address or command	

KEY		
octal	25 PPU	GPIB code
	NAK	ASCII character
hex	15 21	decimal

*USA Standard Code for Information Interchange

Table of GPIB Interface Messages (extended)

Bits b7 b6 b5 →					0 0 0	① MSG	0 0 1	MSG	0 1 0	MSG	0 1 1	MSG	1 0 0	MSG	1 0 1	MSG	1 1 0	MSG	1 1 1	MSG
↓	b3 ↓	b2 ↓	b1 ↓	COLUMN → ROW ↓	0		1		2		3		4		5		6		7	
0	0	0	0	0	NUL		DLE		SP		0		@		P				p	
0	0	0	1	1	SOH	GTL	DC1	LLO	!		1		A		Q		a		q	
0	0	1	0	2	STX		DC2		"		2		B		R		b		r	
0	0	1	1	3	ETX		DC3		#		3		C		S		c		s	
0	1	0	0	4	EOT	SDC	DC4	DCL	$		4		D		T		d		t	
0	1	0	1	5	ENQ	PPC	NAK	PPU	%		5		E		U		e		u	
0	1	1	0	6	ACK		SYN		&		6		F		V		f		v	
0	1	1	1	7	BEL		ETB				7		G		W		g		w	
1	0	0	0	8	BS	GET	CAN	SPE	(		8		H		X		h		x	
1	0	0	1	9	HT	TCT	EM	SPD	)		9		I		Y		i		y	
1	0	1	0	A	LF		SUB		*		:		J		Z		j		z	
1	0	1	1	B	VT		ESC		+		;		K		[		k		{	
1	1	0	0	C	FF		FS		,		<		L		\		l		|	
1	1	0	1	D	CR		GS		–		=		M		]		m		}	
1	1	1	0	E	SO		RS		.		>		N		^		n		~	
1	1	1	1	F	SI		US		/		?	UNL	O		—	UNT	o		DEL	

②

Address command group (ACG)

Universal command group (UCG)

Listen address group (LAG)

Talk address group (TAG)

Primary command group (PCG)

Secondary command group (SCG)

Notes:

①MSG=INTERFACE MESSAGE (Sent by ATN of True: Low level.)
②b1=DI 01……b7=DI 07 (b1 through b7 correspond to DI01 to DI07 sequence.)

GTL	Go to Local
SDC	Select Device Clear
PPC	Parallel Poll Configure
GET	Group Execute Trigger
TCT	Take Control
LLO	Local Lockout
(ACG)	Addressed Command Group
(UCG)	Universal Command Group
(LAG)	Listen Address Group
(TAG)	Talk Address Group
(PCG)	Primary Command Group
(SCG)	Secondary Command Group
DCL	Device Clear
PPU	Parallel Poll Unconfigure
SPE	Serial Poll Enable
SPD	Serial Poll Disable
UNL	Unlisten
UNT	Untalk

Table of Interface Message group

DI08	DI07	DI06	DI05	DI04	DI03	DI02	DI01	Interface message group (G)
×	0	0	0	b4	b3	b2	b1	Addressed command G
×	0	0	1	b4	b3	b2	b1	Universal command G
×	0	1	b5	b4	b3	b2	b1	Listen address G
×	0	1	1	1	1	1	1	Unlisten (UNL)
×	1	0	b5	b4	b3	b2	b1	Talker Address G
×	1	0	1	1	1	1	1	Untalk (UNT)
×	1	1	b5	b4	b3	b2	b1	Secondary command G

Table of Address Assignmemts

Address character		Address swich setting					Primary address	Factory address set device
Talk	Listen	5	4	3	2	1		
b_7 b_6	b_7 b_6	b_5	b_4	b_3	b_2	b_1		
1 0	0 1	↓	↓	↓	↓	↓	10 Decimal	
@	SP	0	0	0	0	0	0	
A	!	0	1	0	0	1	1	
B	"	0	0	0	1	0	2	
C	#	0	0	0	1	1	3	
D	$	0	0	1	0	0	4	
E	%	0	0	1	0	1	5	
F	&	0	0	1	1	0	6	
G	'	0	0	1	1	1	7	
H	(	0	1	0	0	0	8	
I	)	0	1	0	0	1	9	
J	*	0	1	0	1	0	10	
K	+	0	1	0	1	1	11	
L	,	0	1	1	0	0	12	
M	-	0	1	1	0	1	13	Printer
N	.	0	1	1	1	0	14	Plotter
O	/	0	1	1	1	1	15	
P	0	1	0	0	0	0	16	
Q	1	1	0	0	0	1	17	
R	2	1	0	0	1	0	18	
S	3	1	0	0	1	1	19	
T	4	1	0	1	0	0	20	
U	5	1	0	1	0	1	21	
V	6	1	0	1	1	0	22	
W	7	1	0	1	1	1	23	
X	8	1	1	0	0	0	24	
Y	9	1	1	0	0	1	25	
Z	:	1	1	0	1	0	26	
[	;	1	1	0	1	1	27	
\	<	1	1	1	0	0	28	
]	=	1	1	1	0	1	29	
∧	>	1	1	1	1	0	30	
?	—	1	1	1	1	1	31	UNL,UNT

APPENDIX C

COMPARISON TABLE OF CONTROLLER'S GPIB INSTRUCTIONS

Function	Controller				
	PACKET V	PC9800	IBM-PC (NI-488.2)	IBM-PC (NI-488)	HP9000 series
Outputs data to a device	WRITE @ device number: data	PRINT @ listener address; data	CALL Send( )	CALL IBWRT( )	OUTPUT device selector; data
Output binary data to a device	BIN WRITE @ device number: data	WBYTE command; data	CALL SEND Cmds( )		
Assigns data entered from a device to a variable	READ @ device number: variable	INPUT @ talker address, listener address; variable LINE INPUT @ talker address, listener address; variable	CALL Receive( )	CALL IBRD( )	ENTER device selector; variable
Assigns binary data entered from a device to a variable	BIN READ @ device number: variable	RBYTE command; variable			
Initializes an interface	IFC @ select code	ISET IFC	CALL Send IFC( )	CALL IBSIC( )	ABORT select code
Turns REN line on	REN @ select code	ISET REN	CALL Enable Remote( )	CALL IBSRE( )	REMOTE device selector (select code)
Turns REN line off	LCL @ select code (sets all devices local) LCL @ device number (sets only specified devices to listeners, and sends out GTL command)	IRESET REN	CALL Enable Local( )	CALL IBSRE( ) CALL IBLOC( )	LOCAL device selector (select code) LOCAL device selector (select code + primary address)
Outputs interface message(s) and data	COMMAND @ select code: Character string for message [;data]			CALL IBCMD( ) CALL IBCMDA( ) (asynchronous)	SEND select code; message string
Triggers a specified device	TRG @ device number	WBYTE & H3F, listener address, secondary address, &H08;	CALL Trigger( )	CALL IBTRG( )	TRIGGER device selector

Function	Controller				
	PACKET V	PC9800	IBM-PC (NI-488.2)	IBM-PC (NI-488)	HP9000 series
Initializes devices	CDL @ select code (all devices having a specified select code) DCL @ device number (specified devices only)	WBYTE &H3F, &8H14; WBYTE &H3F, listener address, secondary address, &H04	CALL DevClear( )	CALL IBCLR( )	CLEAR device selector (select code) CLEAR device selector (select code + primary address)
Prevents a device from being switche d over from remote to local	LLO @ select code	WBYTE &H3F, &H11;	CALL SendLLO( ) CALL SetRWLS( )	LOCAL LOCKOUT	
Transfers control to a specified device	RCT @ device number	WBYTE talker address, &H09;	CALL Pass Control( )	CALL IBPCT( )	PASS CONTROL
Sends out a service request	SRQ @ select code	ISET SRQ		CALL IBRSV( )	REQUEST select code
Performs serial polling	STATUS @ device number	POLL	CALL Read Status Byte( ) CALL AllSpoll( )	CALL IBRSP( )	SPOLL (device selector) (function)
Sets a terminator code	TERM IS	CMD DELIM		CALL IBEOS( ) CALL IBEOT( )	
Sets a limit value for checking a time-out		CMD TIMEOUT		CALL IBTOM( )	
Wait to SRQ			CALL WaitSRQ( )	CALL IBWAIT( )	

MS2665C/67C/68C
Spectrum Analyzer
Operation Manual
Programming
(PTA Control)

TABLE OF CONTENTS

SECTION 1

GENERAL

TABLE OF CONTENTS

SECTION 1
GENERAL

PTA (Personal Test Automation) is the MS2665C/67C/68C spectrum analyzer equipped with a programming language interpreter function to enable programming controls and calculations directly connected with the measurement system with a high-speed language of PTL (Personal Test Language).

In addition to the basic commands similar to BASIC, PTL provides GPIB control commands, file operation commands, screen control commands and function control commands for controlling most functions of the MS2665C/67C/68C.

Programs that can be executed by PTA are two types, including the “PTA program” to be executed by specifying RUN” from the PTA menu, and the “PTA library” to be executed by registering it to the User key menu. Both the PTA program and PTA library are prepared using the universal edit program on an external personal computer, and registered to MS2665C/67C/68C via RS-232C or GPIB. It is also possible to save the edited PTA program/PTA library to a memory card, as a text file, and input it to the memory card interface of MS2665C/67C/68C. Since inputted programs can be stored in the built-in nonvolatile program memory, no efforts otherwise required for reloading after each power-off are necessary.

As for the external interface to PTA, there are GPIB, RS-232C parallel (Centro) and PTA parallel I/O. RS-232C and GIPB connect an external computer to realize PTA to communicate with the computer through a communication memory (dual port memory). PTA is also capable of controlling an automatic inspection system for electronic components or a trimming machine by connecting with such equipment using PTA parallel I/O.

PTA Specifications

The PTA specifications are listed below:

■Display

- Number of displayed characters : 54 characters/line×20 lines (43 characters/line for menu display)
- Displayable characters : Alphabetic upper-and lower-case characters, numerals, special symbols, and cursors
- Character font : 6×12 dots (small type)
- Graphics : Straight line, square, circle and arc
- Screen : 320×240 dots×16 screens

■Input and execution control

- Input : Front panel, and external computer (by RS-232C, GPIB)
- Execution control : Front panel, and external computer (by RS-232C, GPIB)

■Memory

- Program memory : 196 kbytes
- Memory card : 256 kbytes, 512 kbytes, 1 Mbyte, 2 Mbytes

■Language Version PTL - V1.6

- Commands : Edit commands
  Program execution commands
  File commands

- Statements : Basic statements
  GPIB statements
  Event statements
  Dual port statements

- Subroutines : Display subroutines
  Filing subroutines
  GPIB subroutines
  Interface subroutines
  Panel subroutines
  Waveform memory subroutines

• Functions :	Arithmetic functions Boolean functions Statistical functions Character string functions System functions
• Variables :	Up to a maximum of 256 user-defined variables (numeric variable, character string variable) System variables 22 types
• Data types :	Real number: Significant digits =15 digits Exponential = 10^{308} to 10^{-307} Integer –32768 to 32767 Character: 256 characters max. Bit: 8 bits max.

■Interfaces

- RS-232C
- GPIB
- Parallel (centronics) (option 10)
- PTA parallel I/O (option 14) (MS2665C only)

Output port A	8 bits
Output port B	8 bits
I/O port C	4 bits
I/O port D	4 bits
Control port	3 bits

PTL Command of PTA

Table 1-1 shows the PTL (Personal Test Language) commands provided with the PTA :

Table 1-1 PTL Command of PTA

Item	Format
Edit Commands	
Program input	Line number statement
Copy	PCOPY new start-line number, [increment], copy-source start-line number, copy-source stop-line number
Delete	DELETE [start-line number][,[stop line number]] or [line number] [RETURN]
Renumber	RENUM [new start-line number[,increment[, old start-line number [old stop-line number]]]]
List output (CRT)	LIST [start-line number] [,[stop-line number]
List output (printer)	LISTG address [,start-line number] [,[stop-line number]]
Program size	PMEMO
Execute commands	
Program execution start	[RUN] menu key or RUN [start-line number] [, suspension-line number]
Suspension of program execution	[STOP] menu key
Continuation of suspended program execution	[CONT] menu key, CONT [suspension-line number]
Discontinuation of program execution	[RESET] menu key
Direct execution	Statement [RETURN]
File commands	
Save file	SAVE program name [, start-line number [, stop-line number]]
Load file	LOAD program name
Overlay	OVERLAY
File list display	[PLIST] menu key
Delete file	PDEL Program name
Start-up registration	STARTP program name or STARTP @
Start-up cancel	CANCEL or CANCEL @

Table 1-1 PTL Command of PTA (Continued)

Item	Format
Statements	
Comments	REM ["comment"] or 'comment
Array declaration	DIM array variable
Assignment	[LET] variable = expression (functions,variables or constants)
Jump	GOTO line number or GOTO *label
Jump to subroutines	GOSUB line number or GOSUB *label
Return from subroutines	RETURN
Decision	IF condition statement
Loop beginning	FOR numeric variable = initial value TO ending value STEP step value
Loop end	NEXT numeric variable
Key input	INPUT ["display character string",] variable [, variable...]
Display	PRINT variable [: format][, variable [: format]...][;]
Reverse display	PRINTR variable [: format][, variable [: format]...][;]
GPIB input	READ address, input variable [, variable...]
GPIB input (1 byte)	BREAD address, input variable [, variable...]
GPIB input (2 bytes)	WREAD address, input variable [, variable...]
GPIB output	WRITE address, variable [: format][;]
GPIB output (1 byte)	BWRITE address, variable [: format]
GPIB output (2 bytes)	WWRITE address, variable [: format]
Set measurement parameter	PUT string variable (or string)
Read measurement parameter (1)	GET string variable (or string), input variable
Read measurement parameter (2)	COM character string variable (or character constant)> input variable
Wait	WAIT time (unit is second, minimum 0.01 s.)
Subroutine call	CALL subroutine name
Cursor location (home position)	HOME
Cursor location	LOCATE (X, Y)
Erase screen	ERASE
Program end	STOP
Display error	Line NO␣SOS␣ "Grammer error expression"
Jump on Error	ERROR (error number, line number or * label)
Error main	ERRMAIN
Return to main routine	RETMAIN

Table 1-1 PTL Command of PTA (Continued)

Item	Format
Statement (cont'd)	
Initialization of variable	CLEAR
Data statement	DATA constant [, constant..., constant]
Specification of input data statement	RESRORE [line number or ∗ label]
Data-statement input	RDATA variable [, variable...]
Program reading/execution	CHAIN "file name"
Register an error interrupt routine	ON ERROR line number or ∗ label
Unregister an error interrupt routine	OFF ERROR
Return from an error interrupt routine	RETERR RETRY RESUME line number or ∗ label GIVEUP
Register an event interrupt routine	ON EVENT I/O number, line number or ∗ label
Enable an event interruption	ENABLE EVENT I/O number, event 3, event 2, event 1, event 0
Disable an event interruption	DISABLE EVENT I/O number [, event 3, event 2, event 1, event 0
Return from an event	RETINT
Character size specification	DCHSIZE character size number
Pseudorandom number string setting	RNDMIZE
Dual-port-memory statement	
Write data	WDPM memory No., variable [: format]
Read data	RDPM memory No., input variable

Table 1-1 PTL Command of PTA (Continued)

Item	Format
Screen subroutines (cont'd)	
Screen subroutines	
Displayed-item erasure	CALL CER(M)
Displayed-item restoration	CALL CRN(M)
Screen erasure	CALL CFL(M)
Character string display	CALL DCH(X,Y,text,M[,N])
Straight-line display	CALL DLN(XØ,YØ,X1,Y1,M[,N])
Square display	CALL DRC(XØ,YØ,X1,Y1,M[,N])
Circle display	CALL DCR(X,Y,R,M[,N])
Arc display	CALL DAR(XØ,YØ,RØ,W1,W2,M[,N])
Soft key label registration	CALL DEF(M,text)
Filing subroutines	
Open a file (read)	CALL OPNI Character string variable (or character constant)
Open a file (write)	CALL OPNO Character string variable (or character constant)
Delete a file	CALL FDEL Character string variable (or character constant)
Load data	CALL DALD variable
Save data	CALL DASV variable
Close a file	CALL CLS
Panel subroutines	
Lock front-panel key operation	CALL PNLL(Ø)
Unlock front-panel key operation	CALL PNLU(Ø)
Waveform memory subroutine	
Copy memory	CALL COPY(MØ,M1)
Data conversion	CALL CONV(K,MØ,M1,PØ,P1[,D])
Frequency axis logarithm conversion	CALL SWLG(K,MØ,M1)
GPIB subroutine	
Interface clear (switching to system controller port)	CALL IFC
Service request	CALL RSV(M)
Take controller	CALL TCT(M)
Switching to device port	CALL DEV
Interface subroutine	
Status byte read	CALL GST(port No., address, input variable)
Interface control	CALL GPIB(port No., control item No.)
Function	
Arithmetic functions	SIN, COS, TAN, ASN, ACS, ATN, LN, LOGEXP, SQR, ABS, SGN, INT, ROUND, DIV, FIX
Boolean functions	NOT,AND,OR,EOR
Character string functions	CHR, VAL, HVAL, BVAL, ASC, CHR$, CVI, CVD, MKI$, MKD$, STR$, HEX$, OCT$, BIN$, INSTR, LEFT$, MID$, RIGHT$, STRING$, LEN, SLEN, SGET$

Table 1-1 PTL Command of PTA (Continued)

Item	Format
Function (cont'd)	
Statistical functions	max, min, sum, mean, var, sta
Dedicated functions	ERRREAD, STATUS, DTREAD$, RND
System variable	EX0, EX1, EX2, EX3, EX4, EX5, EX6, DTØ, DT1, DT2, DT3, DT4, XMA, XMB, XMG, XMT, XMT, SMA, SMB, SMT, IMA, IMB, RMA, RMB, IOA, IOB, IOC, IOD, EIO
System function	
Maximum value	MAX (M, PØ, P1, N)
Minimum value	MIN (M, PØ, P1, N)
Frequency measurement 1	BNDL (M, PØ, L, N)
Frequency measurement 2	BNDH (M, PØ, L, N)
Frequency measurement 3	MESL (M, PØ, L, N)
Frequency measurement 4	MESH (M, PØ, L, N)
Ripple 1	RPL1 (PØ, P1, N [,R])
Ripple 2	RPL2 (PØ, P1, N [,R])
Ripple 3	RPL3 (PØ, P1, N [,R])
Peak 1	PEKL (M, PØ, L, N [,R])
Peak 2	PEKH (M, PØ, L, N [,R])
Poll 1	POLL (M, PØ, L, N [,R])
Poll 2	POLH (M, PØ, L, N [,R])
Maximum 1	PLRH (M, PØ, N [,R])
Maximum 2	PLLH (M, PØ, N [,R])
Minimum 1	PLRL (M, PØ, N [,R])
Minimum 2	PLLL (M, PØ, N [,R])
Index point frequency	PFRQ (PØ)
Sum	SUM (PØ, P1, N)
Adding search 1	PSML (M, PØ, L, N)
Adding search 2	PSMH (M, PØ, L, N)
Judgment 1	DPOS (M, PØ, P1, N1, N2)
Judgment 2	DNEG (M, PØ, P1, N1, N2)

External Interfaces of MS2665C/67C/68C

MS2665C/67C/68C provides an RS-232C interface and GPIB interface as standard. In addition, a parallel (centronics) interface (option 10) or PTA parallel interface (option 14, MS2665C only) is optionally available. The usage of these interfaces differs by the setting of the connection port.

RS-232C interface

- When the RS-232C interface is selected as the connection port for the external controller (Connect to Controller):

  Connect the device that controls the spectrum analyzer, for example, a host computer. Execution of the PTA program/PTA library is indicated and the PTA program can be interfaced via the dual port memory. Also, the PTA program/PTA library is registered.
- When the RS-232C interface is selected as the connection port to the printer/plotter (Connect to Printer/Plotter):
  By specifying COPY from the PTA program/library, the printer copies the screen.
- When the RS-232C interface is selected as the connection port to the a peripheral device (Connect to Peripheral):

  Serial data transfer is available between the PTA program/library and the external device.

GPIB interface

- When the GPIB interface is selected as the connection port for the external controller (Connect to Controller): In this case, the GP-IB interface enters the devicxe port state. Connect the device that controls the spectrum analyzer, for example, a host computer. Execution of the PTA program/PTA library is indicated and the PTA program can be interfaced via the dual port memory. Also, the PTA program/PTA library is registered.
- When the GPIB interface is selected as the connection port to the printer/plotter (Connect to Printer/Plotter): By specifying COPY from the PTA program/library, the printer copies the screen.
- When the GPIB interface is selected as the connection port to the a peripheral device (Connect to Peripheral): In this case, the GPIB interface works as a system controller port. It is possible to control external devices from the PTA program/library.

Parallel (centronics) interface

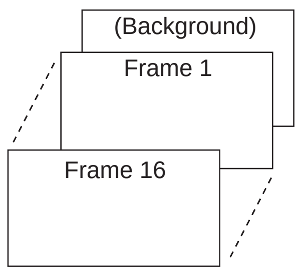
- When the parallel (centronics) interface is selected as the connection port to the printer/plotter (Connect to Printer/Plotter):

  By specifying COPY from a PTA program/library, the printer copies the screen.

Screen Configuration of PTA

This section describes the screen specifications of PTA mounted in the MS2665C/67C/68C.

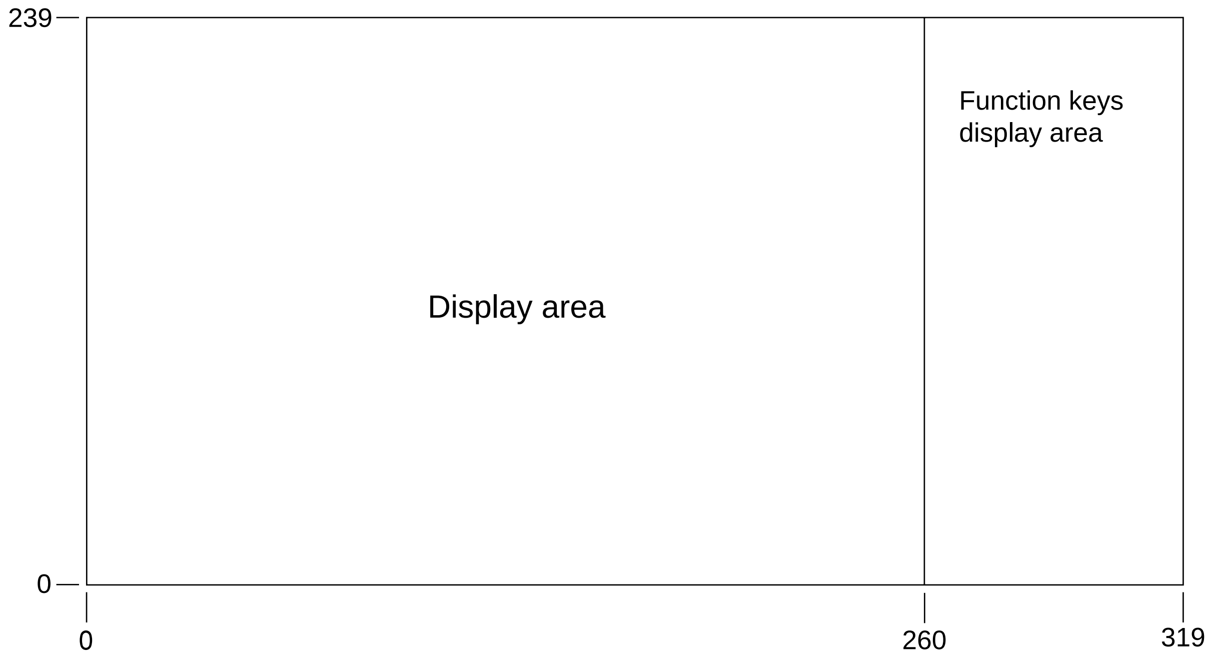
Physical screen configuration

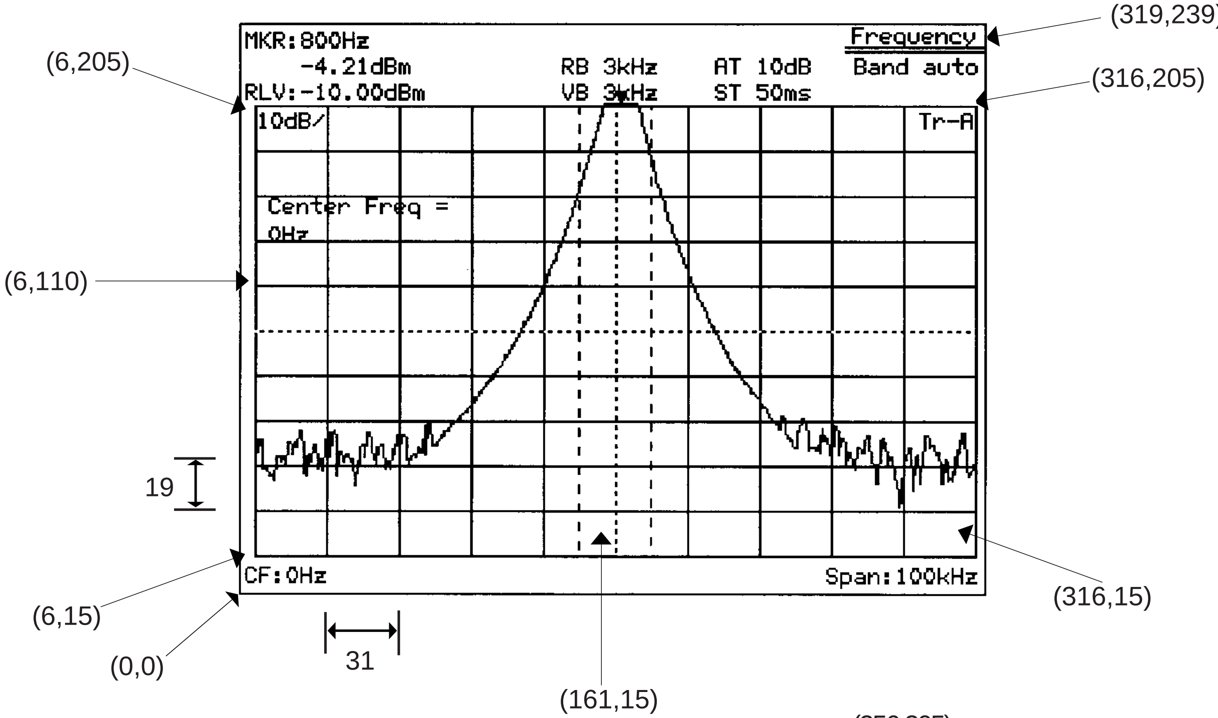


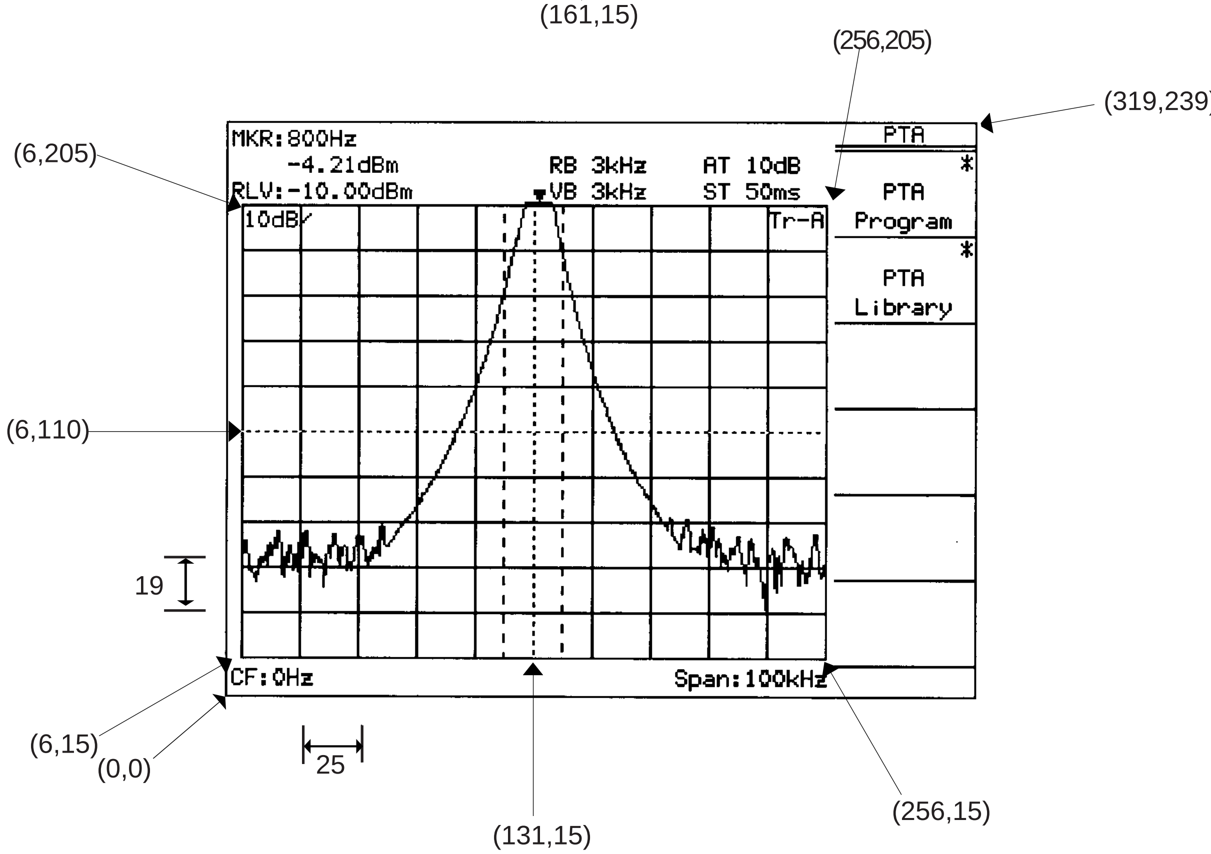
Frame 1 : Waveform display background
2 : Scale lines
3 : Waveform 2
4 : Waveform 1
5 : Parameters (title, reference level, RBW, VBW, center frequency, span, etc.)
6 : Display lines, reference markers
7 : Triggers, indicators
8 : Marker zones
9 : Template/mask standard lines
10 : Multi-marker Nos.
11 : (Not used)
12 : Markers, marker values
13 : PTA screen
14 : Menu background
15 : Menu characters
16 : Setup and parameter characters, error messages

Note: This frame number is controlled inside the spectrum analyzer. The number is different from the number used by screen subroutines such as CALL CFL.

Display form




MKR:800Hz
-4.21dBm
RLV:-10.00dBm
RB 3kHz
VB 3kHz
AT 10dB
ST 50ms
Frequency
Band auto
10dB/
Tr-A
Center Freq =
0Hz
CF:0Hz
Span:100kHz
(319,239)
(6,205)
(316,205)
(6,110)
19
(6,15)
(0,0)
31
(161,15)
(316,15)


(256,205)
(319,239)
MKR:800Hz
-4.21dBm
RLV:-10.00dBm
RB 3kHz
VB 3kHz
AT 10dB
ST 50ms
PTA
PTA
Program
PTA
Library
10dB/
Tr-A
CF:0Hz
Span:100kHz
(6,205)
(6,110)
19
(6,15)
(0,0)
25
(131,15)
(256,15)

SECTION 2

PTA OPERATION

TABLE OF CONTENTS

SECTION 2
PTA OPERATION

Outlining the Operation

PTA of MS2665C/67C/68C is capable of executing/operating two types of automation programs, the "PTA program" and "PTA library".

PTA program :

One program can be loaded and executed on the execution memory (RAM) of MS2665C/67C/68C.

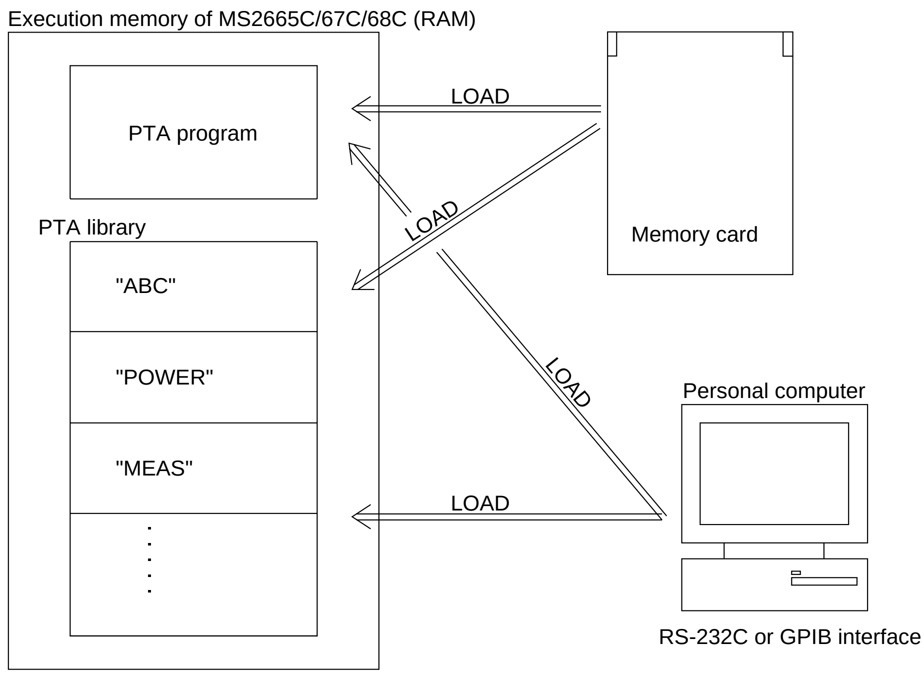
A PTA program is loaded and executed on menus following [SHIFT] + [PTA]→[PTA Program : F1].

This function is the same as the PTA functions and PTA program execution provided in the existing measuring instruments of our make (for example, MS2601B, MS2602A, MS8604A, etc.).

PTA library :

Multiple programs can be loaded and executed on the execution memory (RAM) of MS2665C/67C/68C.

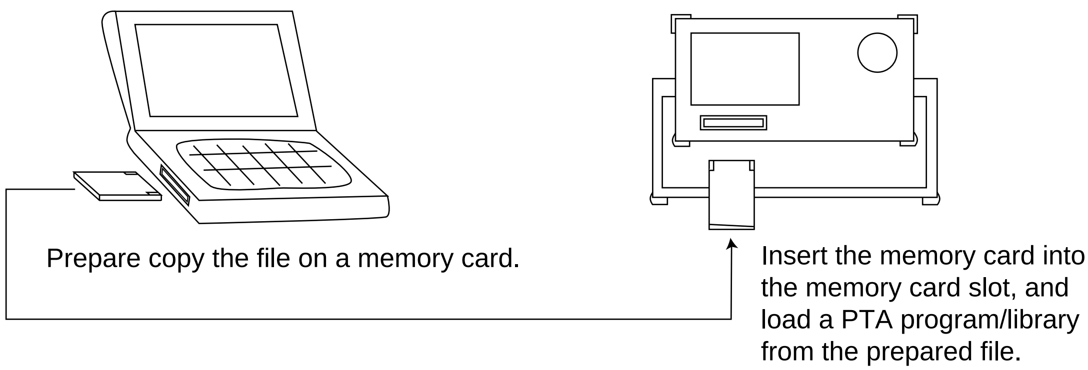
A PTA library is loaded and executed on menus following the [SHIFT] + [PTA]→[PTA Library : F2] keys. The PTA library can be executed by registering it to a menu of the [User] key and pressing the appropriate Fkey.
Also, the PTA library can be executed by directly inputting the PTA library name as a remote control command from the controller.



A PTA program or PTA library can be loaded to the execution memory of MS2665C/67C/68C by either of the following three methods:

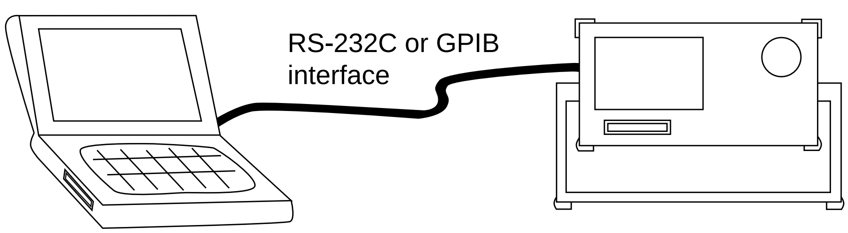
(1) Prepare a PTA program/library as a text file of in DOS format on a memory card, and load it to spectrum analyzer.

- Prepare the PTA program/library file using the edit program (editor) on the personal computer.
- Copy the prepared file to the memory card.
- Insert the memory card to the memory card slot of spectrum analyzer, and load it from the operation menu of the PTA program or PTA library.



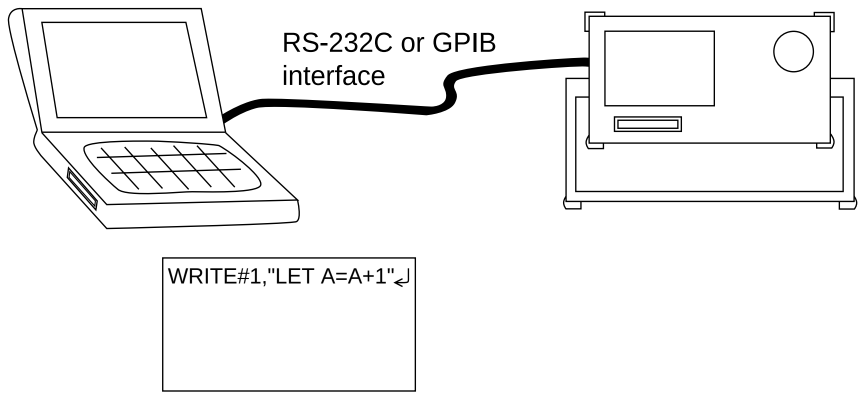
(2) Prepare a PTA program/library file on the personal computer, and load it to spectrum analyzer via the RS-232C or GPIB interface.

- Prepare the PTA program/library file using the edit program (editor) on the personal computer.
- Load the data (PTL statement) of the prepared file to spectrum analyzer via the RS-232C or GPIB interface.



(3) Remote-controlling spectrum analyzer from the personal computer, directly input the PTL statement.

- Remote-control spectrum analyzer from the personal computer via the RS-232C or GPIB interface and get the PTA operation screen.
- Sending a PTA statement line by line to spectrum analyzer, prepare a PTA program/library on the execution memory of spectrum analyzer.



Operations Related to PTA Program

Operations related to the loading and execution of PTA programs are described below. Operations are the same as those of the PTA functions and PTA program execution provided in the existing measuring instruments of our make (for example, MS2601B, MS2602A, MS8604A, etc.).

Startup of PTA

PTA is actuated by pressing the [SHIFT] + [PTA : 7] keys on the front panel or inputting the remote control command "PTA_1". The screen is erased and the cursor appears at the home position (top left of the screen).

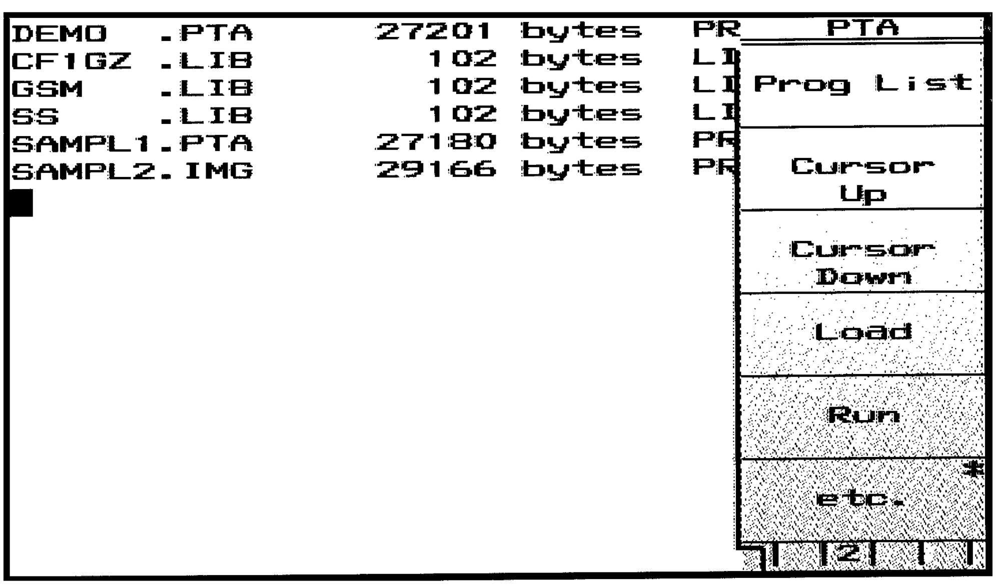
Additionally, by registering a PTA program/library as a startup program, it can be actuated and executed upon powering on. (For details about the startup registration of the PTA program, see Section 3 "STARTUP command". Likewise, for details about the PTA library, see Section 3 "POWERUP command".)

Loading the PTA program from memory card

A PTA program can be prepared as a text file in DOS format on a memory card and loaded to spectrum analyzer by the edit program (editor) of the personal computer and the like.

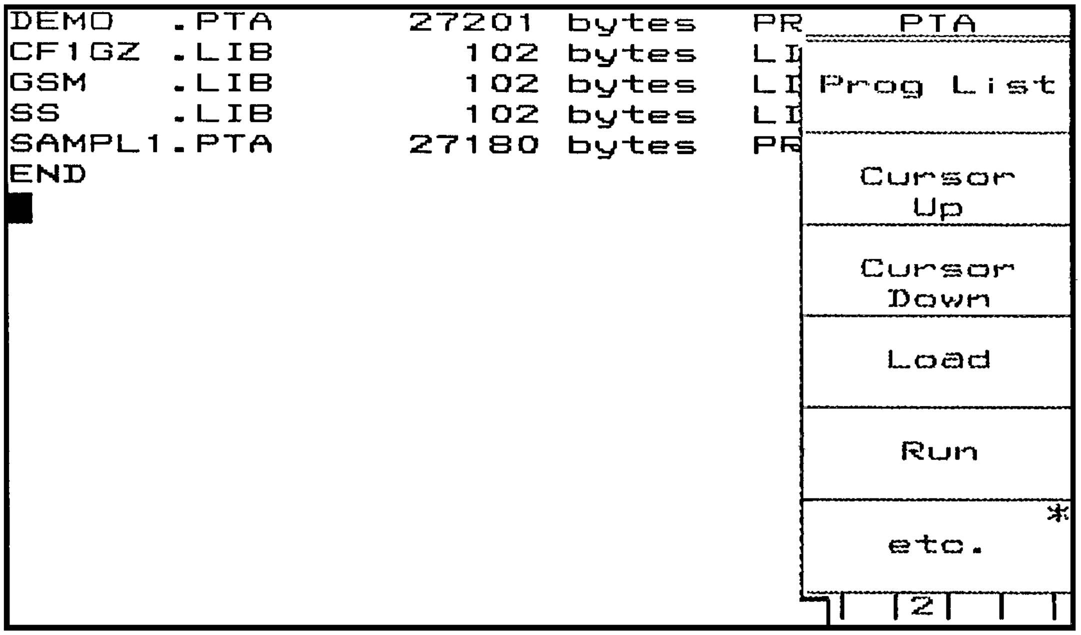
(1) Press [SHIFT] + [PTA : 7] → [PTA Program : F1] keys and get the PTA program operation mode (PTA ON).

(2) Press the [PLIST : F1] key of the PTA program menu (page 2) to display a list of program names stored in the memory card.



(3) Press [CURSOR UP : F2] and [CURSOR DOWN : F3] keys and move the cursor to the program name to load.

(4) Press the [LOAD : F4] key.
Read out the PTA program from the memory card. When reading is completed, the [END] message is displayed.



(5) Press the [RUN : F5] key to execute the program.

(6) To stop execution, press the [RESET : F4] key of the PTA program menu (page 1).

Execution, stop of the PTA program

After loading a PTA program from a memory card, the PTA program can be executed and stopped without loading operation. Since the execution memory of the PTA program is backed up by batteries, it is retained under the loaded condition after powered off. Condition under execution is not retained.

(1) Press [SHIFT] + [PTA : 7]→[PTA Program : F1] keys and get the PTA program operation mode (PTA ON).

(2) Press the [RUN : F1] key of the PTA menu (page 1) to execute the program.

(3) To interrupt program execution, press the [STOP : F2] key.

(4) To resume program execution, press the [CONT : F3] key.

(5) To stop program execution, press the [RESET : F4] key. To restart execution, press the [RUN : F1] key.

PTA termination

To terminate PTA, press the [RESET : F4] key to stop program execution, and then press the [PTA OFF : F5] key or input a remote control command "PTA␣0".

Afterwards, the screen (which has been displayed by display subroutine) is cleared to be returned to ordinary measurement screen.

Note

For the display subroutine, see Section 5, "System Subroutines".

Format of PTA program file

There are two formats for a PTA program file on a memory card, as follows:

(1) Text format

The extender for a PTA program file in text format is ".PTA". An example of the PTA program file in text format is shown below.

```
10 '=========================================================
20 '== MS2660 series PTA Program/Library Sample Program   ===
30 '=========================================================
40 '
50 HOME&ERASE'                        Erase PTA screen
60 PRINT "   Hello PTA World!!"'      Print message
70 PUT   "IP"'                        Preset MS2660 series
80 PUT   "CF 100MHZ"'                 Set center frequency 100MHz
90 PUT   "SP 100KHZ"'                 Set frequency span 100kHz
100 PUT  "MKPK"'                      Perform peak search
110 STOP'                             Stop execution
```

(2) Execution format

The extender of a PTA program file in execution format is ".IMG". The PTA program file in execution format is stored in the form of binary data and cannot be edited on the personal computer.

The file in execution format can be prepared by adding ".IMG" as the extender to the file name by the LOAD command of PTA. Storing the file in execution format will reduce loading time.

Operations Related to PTA Library

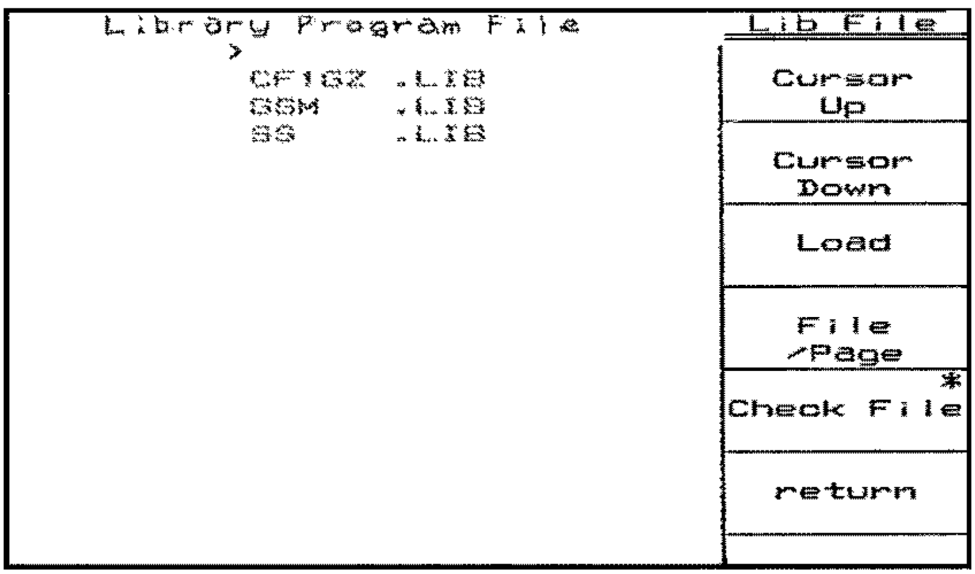
Operations related to the loading and execution of the PTA library are described below.

Loading the PTA library from memory card

A PTA library can be prepared as a text file in DOS format on a memory card and loaded to spectrum analyzer by the edit program (editor) of the personal computer and the like.

(1) Press [SHIFT] + [PTA : 7] → [PTA Library : F2] keys and get the PTA library operation mode (PTA ON).

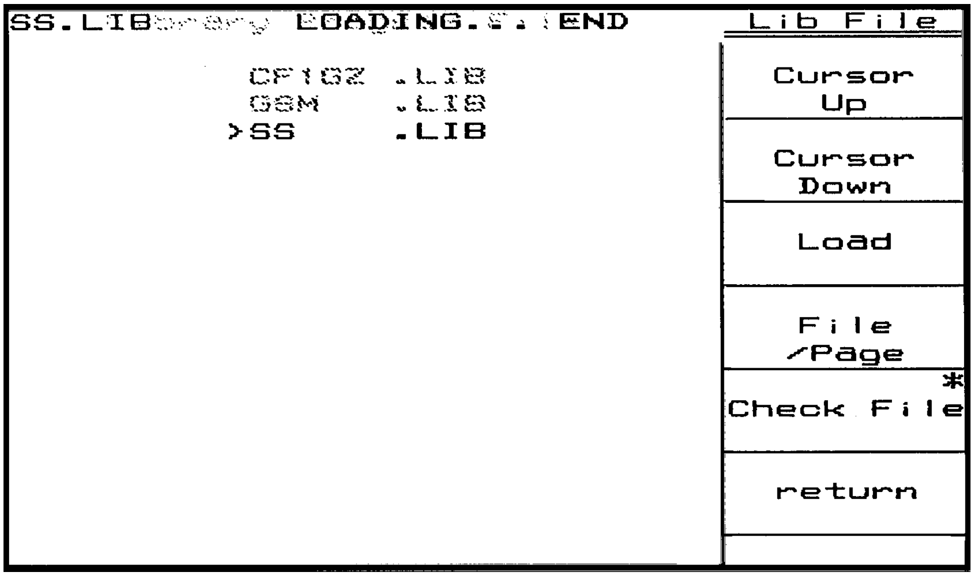
(2) Press the [Library File : F2] key of the PTA library menu to display a list of library files stored in the memory card. If the list cannot be displayed at a time, press the [File/Page : F4] key to display the next page.



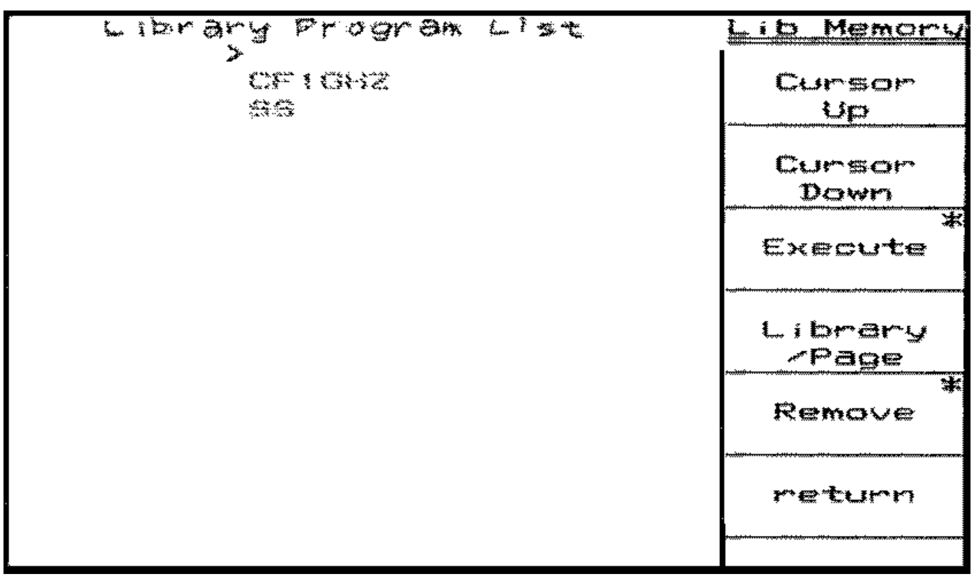
(3) Press [CURSOR UP : F1] and [CURSOR DOWN : F2] keys and move the cursor to the library file name to load.

(4) Press the [LOAD : F3] key.

Read out the PTA library from the memory card. When reading is completed, the [LOADING...END] message is displayed.



After loading, the PTA library loaded on the execution memory can be displayed in list form by pressing the [Library Memory : F1] key of the PTA library menu.

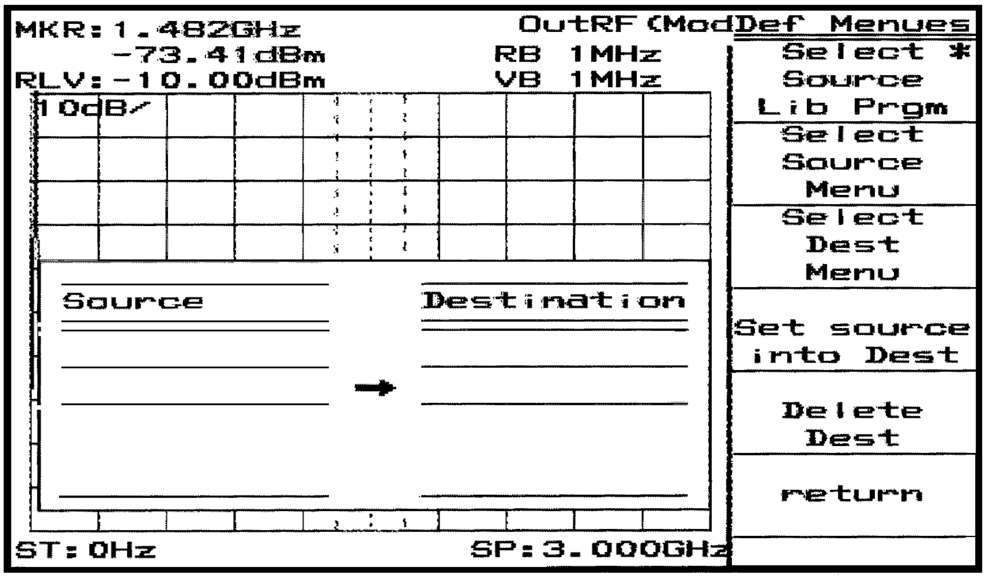


Also test execution can be done by operating menus following the [Executed : F3] key.

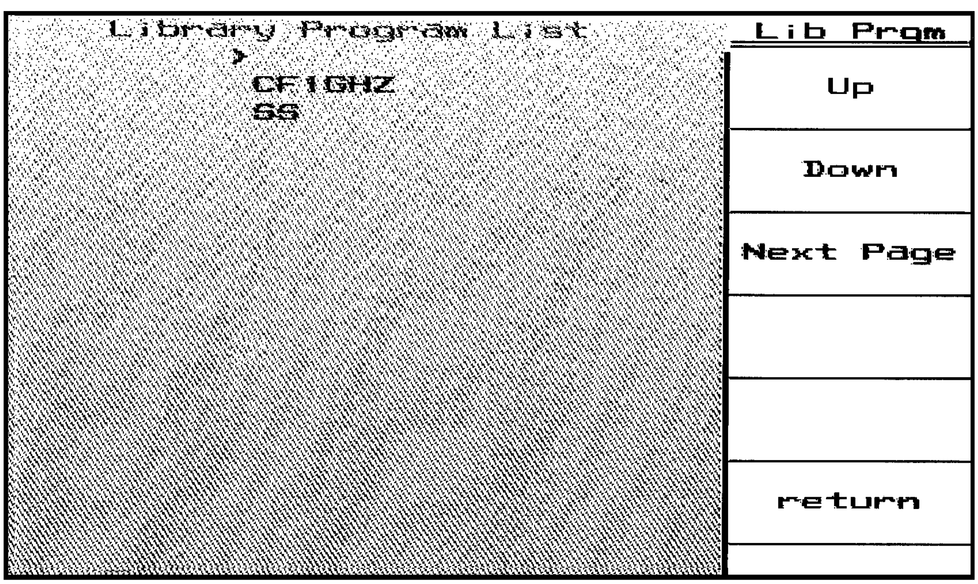
Registering the PTA library to user key

The PTA library loaded to the execution memory can be executed by registering it to a menu of the [User] key on the front panel. The registering operation procedure of the PTA library to the User key described below.

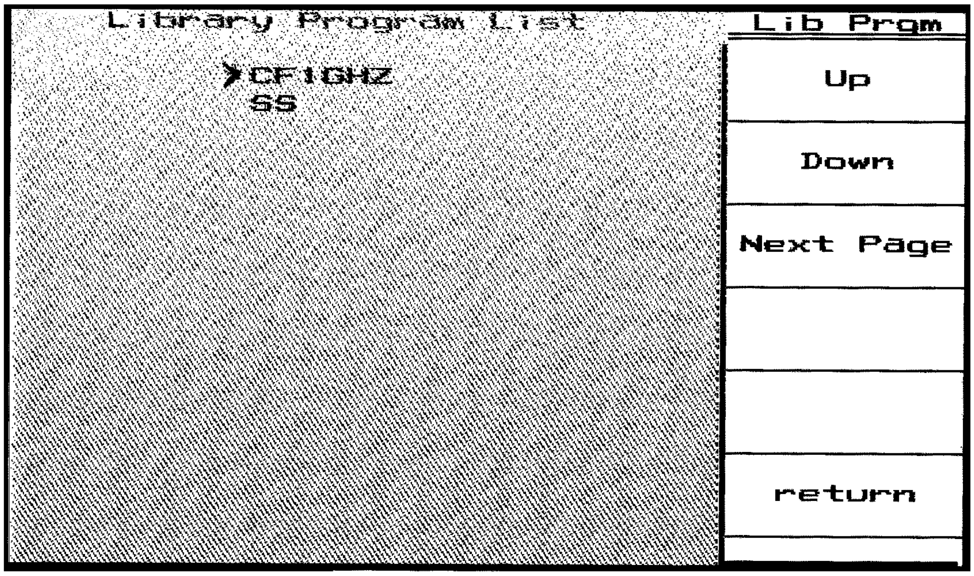
(1) Press [SHIFT] + [User Define : 8] → [Define Menus : F1] keys and display the User key registration screen.



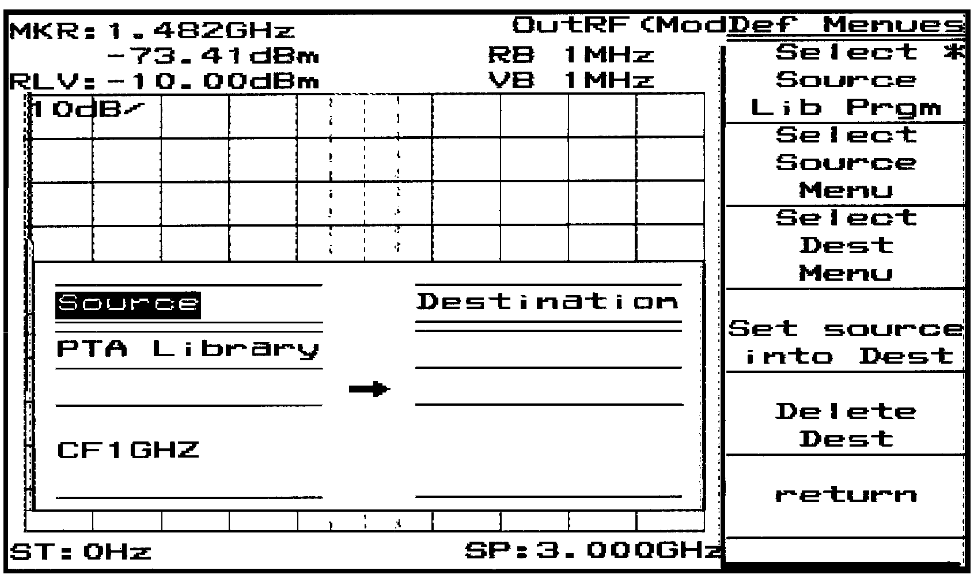
(2) Press the [Select Source Lib prgm : F1] key. The PTA library loaded in the execution memory is displayed in list form.



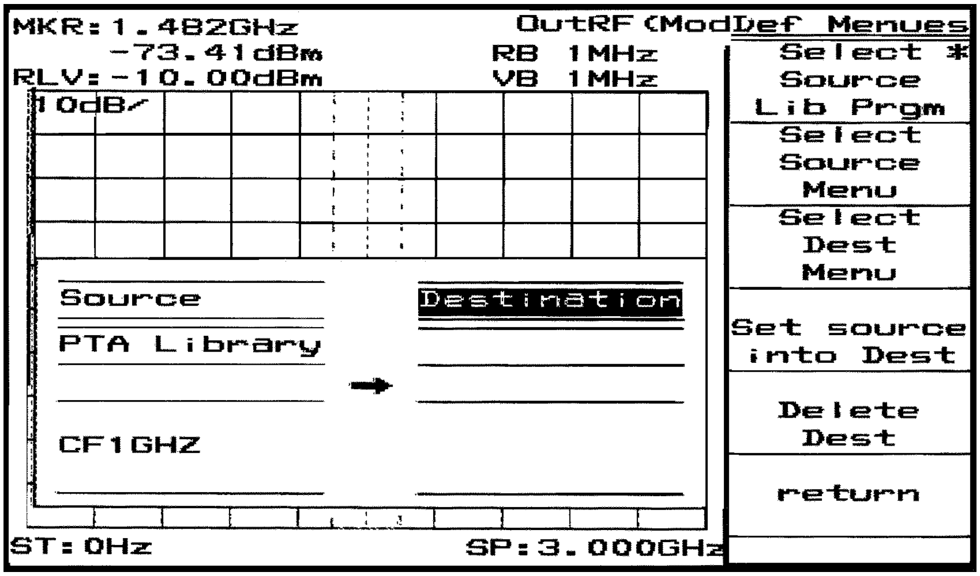
(3) Press [UP : F1] and [DOWN : F2] keys, and move the cursor to the library name to load to the User key.



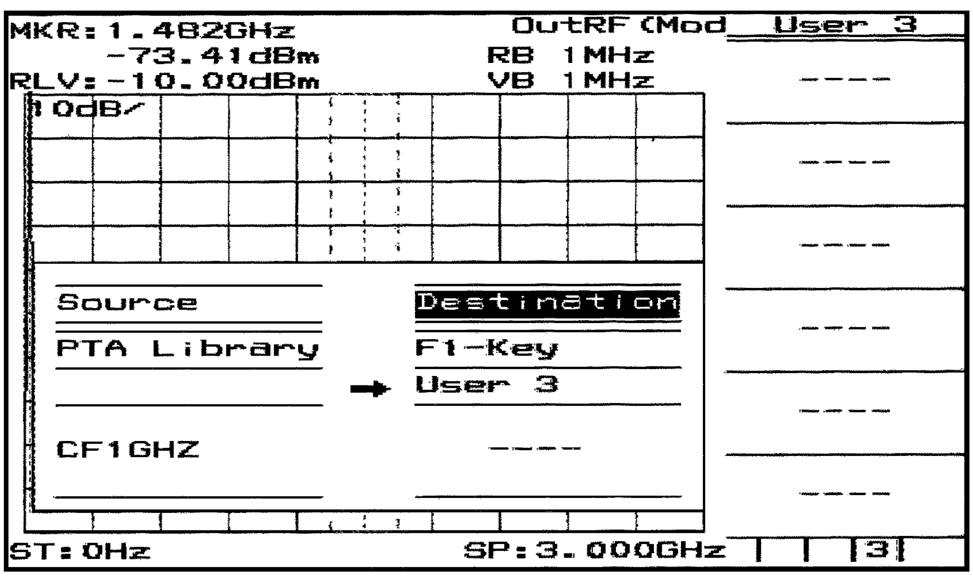
(4) Press the [return : F6] key. The selected library name is displayed in the Source column of the User key registration screen.



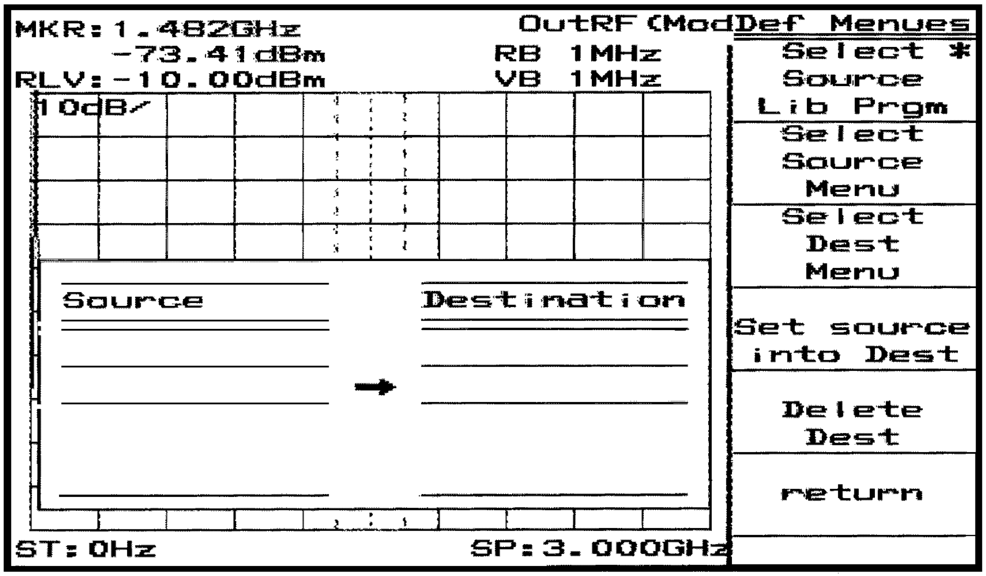
(5) Press the [Select Dest Menu : F3] key. the title in the Destination column of the User key registration screen is inverted, indicating the waiting status for the selection of the destination menu.



(6) Press the [User] key on the front panel and press a menu to register. Each time a menu is pressed, the selected menu is displayed on the Destination column of the User key registration screen. A menu that is pressed last is the destination.

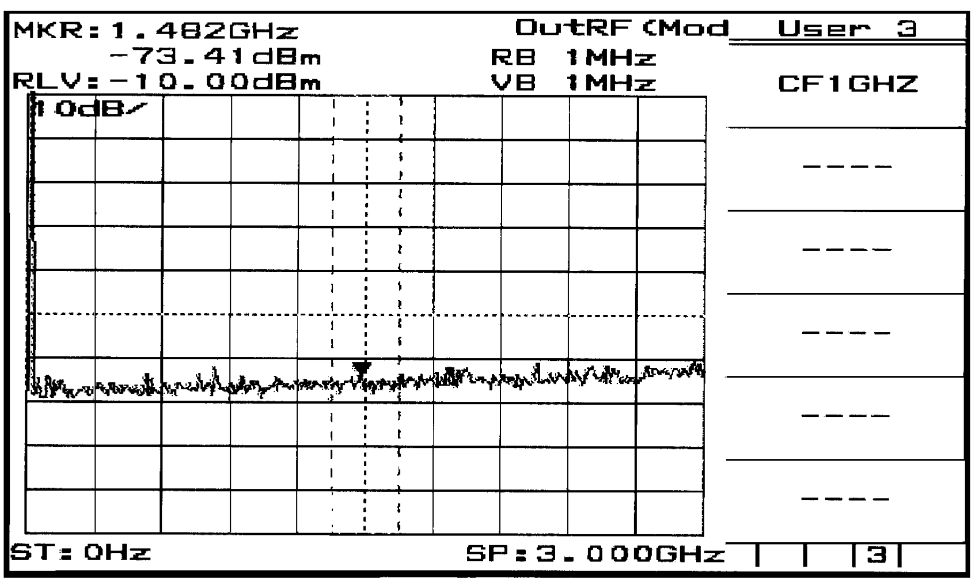


(7) Press [SHIFT] + [User Define : 8] → [Define Menus : F1]→[Set source into Dest : F4] keys to register the execution of the PTA library to the selected User key.



After registering, pressing the [return : F6] key erases the User key registration screen.

Press the [User] key on the front panel and look at the registered menu; the PTA library name is displayed on the menu, indicating that registration is completed.



Press this key to start executing the registered PTA library.

Execution, stop of the PTA library

The PTA library loaded to the execution memory is normally executed by registering it to the User key, but test execution can be done from the PTA library menu.

(1) Press [SHIFT] + [PTA : 7] → [PTA Library : F2] keys and get the PTA library operation mode.

(2) Press the [Library Memory : F1] key and display the PTA library loaded on the execution memory in list form. If the list cannot be displayed at a time, press the [File/Page : F4] key to display the next page.

(3) Press [CURSOR UP : F1] and [CURSOR DOWN : F2] keys and move the cursor to the program name to test-execute.

(4) Press the [Execute : F3] key and get the PTA library test execution mode.

Under the test execution mode, the following operations are available:

(5) Press the [RUN : F1] key to execute the library.

(6) To interrupt library execution, press the [STOP : F2] key.

(7) To resume library execution, press the [CONT : F3] key.

(8) To stop library execution, press the [RESET : F4] key. To restart execution, press the [RUN : F1] key.

Format of PTA library file

There are two formats for a PTA library file on a memory card, as follows:

(1) Text format
The extender for a PTA library file in text format is ".LIA". One PTA library file in text format can store one PTA library only. The title of this PTA library is the same as that of the PTA library file. Data in the PTA library file in text form is totally the same as that of the PTA program, with only an exception of the extender of the file.

(2) Execution format
The extender of a PTA library file in execution format is ".LIB". The PTA program file in execution format is stored as binary data and cannot be edited on the personal computer.
One PTA library file in execution format can store plural PTA libraries. There are no title relations between the PTA library file and PTA libraries stored in it.

Operations related to PTA library

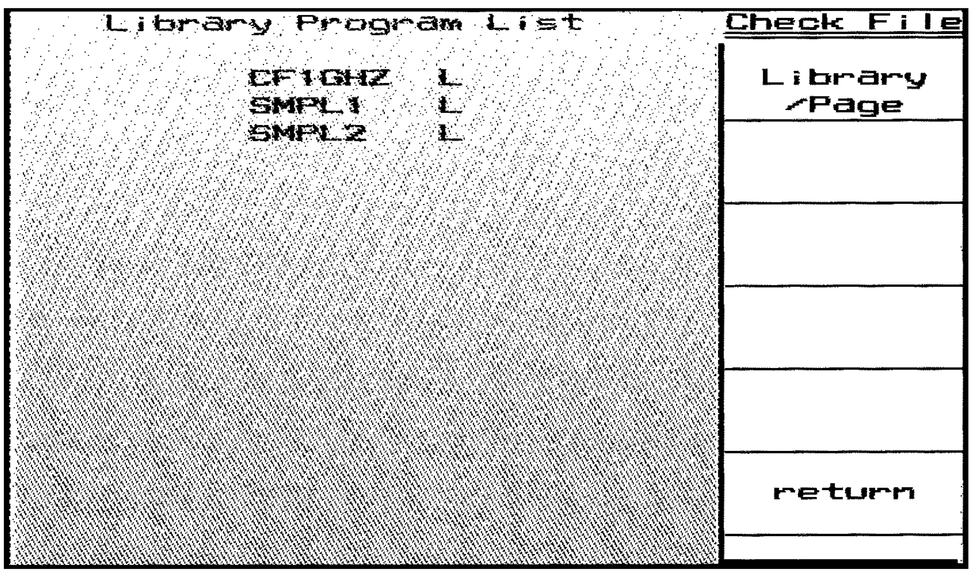
In the case of a PTA library file in execution format, stored PTA libraries cannot be confirmed by a file list. For this purpose, the PTA libraries can be listed by the following operations:

(1) Press [SHIFT] + [PTA : 7] → [PTA Library : F2] keys and get the PTA library operation mode.

(2) Press the [Library File : F2] key of the PTA library menu to display a list of library files stored in the memory card. If the list cannot be displayed at a time, press the [File/Page : F4] key to display the next page.

(3) Press [CURSOR UP : F1] and [CURSOR DOWN : F2] keys and move the cursor to the library file name to confirm PTA libraries stored in it.

(4) Press the [Check File : F5] key.
A list of PTA library files stored in the selected PTA library file is displayed on the screen. If the list cannot be displayed at a time, press the [File/Page : F1] key to display the next page.



Panel Key Operations during PTA Program/Library Execution

Data input keys

The soft keys, numeric keys, and unit keys on the front panel serve as data input keys.

(1) F1, F2, F3, F4 and F5 keys

The F1 to F5 keys are referred to in the program and correspond to the system variables EX1, EX2, EX3, EX4 and EX5 respectively.

Each time the key is pressed, the variable contents are alternately changed to 0 or 1. All the data in these variables are 0 at initial state and resetting. Displayed name in menu can be defined with DEF subroutine.

Note

For EX1, EX2, EX3, EX4 and EX5, see Section 5, " System Variables".

(2) YES and NO keys

These are typing aids for the INPUT statement; the "YES" and "NO" character string can be input by a single key operation.

(3) Numeric keys

These are the [0] to [9] ,[.] and [BS] keys which are used for inputting data on INPUT statement.
Press the [Enter] key to terminate the input; use the [BS] key to delete one character.

(4) Unit keys

Unit key No. 1 : Treats this key as the CR key.
Unit key No. 2 : Treats this key as the [,] key.
Unit key No. 3 : Treats this key as the [–] key.
Unit key No. 4 : Invalid

∗: The figure below shows unit key numbers.

7	8	9		GHz dBm dB	Unit key No. 4
4	5	6		MHz V sec	Unit key No. 3
1	2	3		kHz mV msec	Unit key No. 2
0	.	+/–	Enter	Hz uV usec	Unit key No. 1

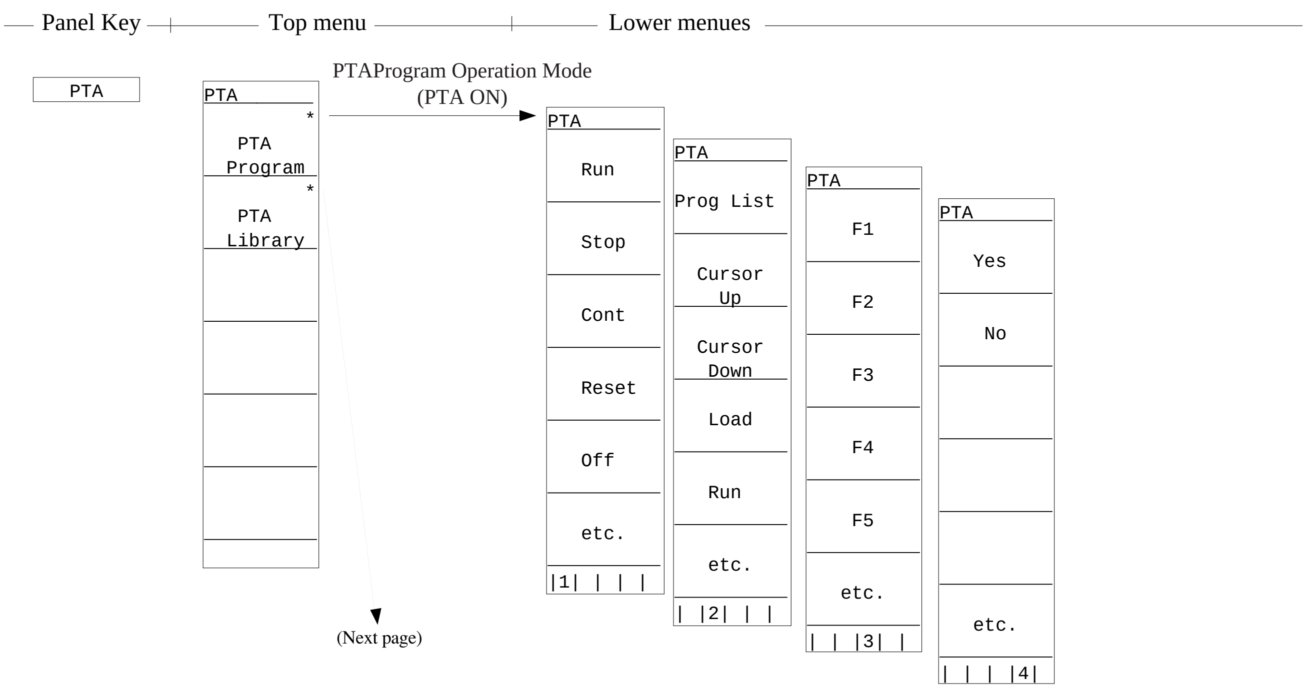
Operation of other panel keys

When PTA is ON, the panel keys are locked-out except for the number/[Enter] keys, [Shift] key, [Local] key and soft keys (F1 to F6).

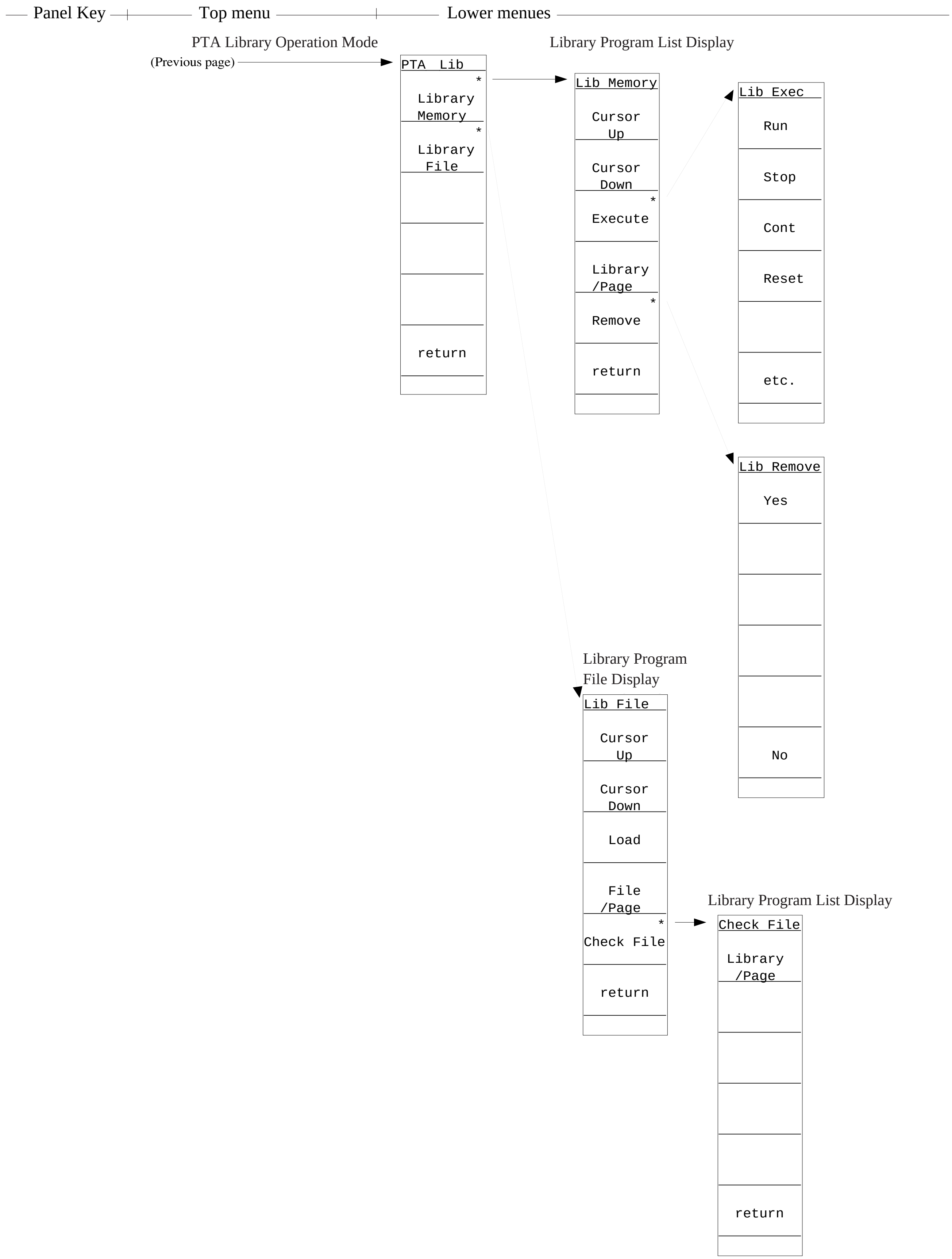
Menu Construction of the PTA Key

Menu layers following [SHIFT] + [PTA : 7] keys are shown below.

Menu Tree



Menu Tree


Panel Key
Top menu
Lower menues
PTA Library Operation Mode
(Previous page)
PTA Lib
*
Library Memory
*
Library File
return
Library Program List Display
Lib Memory
Cursor Up
Cursor Down
*
Execute
Library /Page
*
Remove
return
Lib Exec
Run
Stop
Cont
Reset
etc.
Lib Remove
Yes
No
Library Program File Display
Lib File
Cursor Up
Cursor Down
Load
File /Page
*
Check File
return
Library Program List Display
Check File
Library /Page
return

SECTION 3

PTL COMMANDS

TABLE OF CONTENTS

SECTION 3
PTL COMMANDS

PTL (Personal Test Language) commands include commands for the edition, execution and filing of the PTA programs/libraries, and are composed as shown below:

- PTL commands
  - Edit commands (Program input)
    - PCOPY
    - DELETE
    - RENUM
    - LIST
    - LISTG
    - PAGE SCROLL
    - ROLL UP/DOWN
    - PMEMO
  - Execution Control commands
    - RUN
    - CONT
    - STEP
    - RESET
    - STOP
  - File administration commands
    - SAVE
    - LOAD
    - OVERLAY
    - PDEL
    - PLIST
    - STARTP
    - CANCEL
  - PTA library administration commands
    - EDITLIB
    - EDITPTA
    - RENAME
    - LIBMEM
  - PTA library file administration commands
    - SAVELIB

Program Input Command

(1) Function

When a statement with a line No. is inputted, it is stored as a PTA program/library to the program area. When the line No. is different from those already inputted, the statement is added or inserted, and when the line No. is the same, the statement will replace the already inputted statement.

(2) Format

```
Line number Statement
   |
```
Integer constant from 1 to 65535

Notes:

- When 111 or more characters (including the line number) are input on one line during program input, the program on that line may not be displayed during LIST-command execution after execution of the RENUM command.
- For a description of the RENUM command, see Section 3, "RENUM Command".

PCOPY Command

(1) Function

This statement copies the specified program.
(from <copy-source start-line number> to the <copy-source end-line number>) in the unit of increment specified by <increment> from the <new start-line number>.
If <increment> is omitted, thenÅe10Åfis used as the default value.

(2) Format

```
PCOPY operand 1,[operand 2],operand 3,operand 4
```

New start-line number Increment Copy-source start-line number Copy-source end-line number.

1) `PCOPY 1ØØ,,1Ø,3Ø` Copies the statement (from lines 10 to 30) to location 100 in increments of 10 and labels all sequent.

2) `PCOPY 1ØØ,5,1Ø,3Ø` Copies the statement (from lines 10 to 30) to location 100 and in increment of 5 and labels all sequent.

Notes:

- If the line number of a newly-copied statement is identical to the line number of the current statement, ERROR F101 occurs.
- If a line has more than 111 characters when PCOPY is executed, display is disabled during LIST command execution.

DELETE Command

(1) Function

This command deletes all or part of a program.

(2) Format

```
DELETE [operand 1][,][operand 2]
      operand 1≤operand 2
```

(3) Example

1) `DELETE` Deletes entire program and initializes variable values.
2) `DELETE 1ØØ` Deletes statement on line 100.
3) `DELETE 1ØØ,` Deletes statements on lines 100 to the end line.
4) `DELETE ,5ØØ` Deletes statements on start line to line 500.
5) `DELETE 1ØØ,5ØØ` Deletes statements on line 100 to line 500.

- When deleting only a line, it is possible by Line number [RETURN].

RENUM Command

(1) Function

This command renumbers line numbers used in the program. When the increment value or new line number is omitted, 10 is used as the default value.

(2) Format

```
RENUM [[operand 1][,][operand 2][,][operand 3][,][operand 4]]
```

New-line number   Increment   Start-line number   End-line number

1) `RENUM`	Renumbers all program statements starting from first-line number 10 in increments of 10
2) `RENUM 1ØØ`	Renumbers all program statements from new first-line number 100 in increments of 10
3) `RENUM 1ØØ,5`	Renumbers all program statements starting from new first-line number 100 in increments of 5
4) `RENUM 1ØØ,,5Ø`	Renumbers statements (from line number 50 through the last line) starting from new first-line number 100, in increments of 10
5) `RENUM 1ØØ,5,5Ø`	Renumbers statements (from line number 50 through the last line) starting from new first-line number 100, in increments of 5
6) `RENUM 1ØØ,,,15Ø`	Renumbers statements (from line number 10 through 150) starting from new first-line number 100 in increments of 10
7) `RENUM 1ØØ,,5Ø,15Ø`	Renumbers statements (from line number 50 through 150) starting from new first-line number 100 in increments of 10
8) `RENUM 1ØØ,5,,15Ø`	Renumbers statements (from line number 10 through 150) starting from new first-line number 100 in increments of 5
9) `RENUM 1ØØ,5,5Ø,15Ø`	Renumbers statements (from line number 50 through 150) starting from new first-line number 100 in increments of 5

Notes:

- Labels can be used for operands 1, 3 and 4.
- "ERROR F101" occurs if there is a line number larger than that of operand 4 when operand 1 is smaller than operand 4.
- If the number of characters on a line is more than 111 characters, when the number of lines of the program line becomes two lines or more with RENUM command, ERROR F20 will occur during LIST command execution and display the lines.

LIST Command

(1) Function

This command displays all or part of a program on the LCD.

(2) Format

```
LIST [operand 1][,][operand 2]
          |              |
  Start-line number   End-line number
       operand 1≤operand 2
```

1) `LIST` — Lists entire program
2) `LIST 1ØØ` — Lists the statement on line 100
3) `LIST 1ØØ,` — Lists statements on line 100 to end line
4) `LIST ,5ØØ` — Lists statements on start line to line 500
5) `LIST 1ØØ,5ØØ` — Lists statements on line 100 to 500

Note: Labels can be used for operands 1 and 2.

LISTG Command

(1) Function

This command outputs all or part of a program to a printer connected to the RS-232C/GPIB/parallel (centronics) interface.

(2) Format

```
LISTG address [[,][operand 1][,][operand 2]]
```

Address of printer (0 to 30)

Operand 1 and operand 2 in the LISTG command are used in the same way as the LIST command.

Notes:

- To use RS-232C/GPIB/parallel (centronics) interface from PTA, it is necessary to choose a port to use. The selection of the port, press [SHIFT] + [Interface : .] keys, and then press the [Connect to Peripheral : F6] key several times.
- When the program is output to the RS-232C or parallel (centronics) interface, addresses have no meaning, but they should be specified as a formality.

PMEMO Command

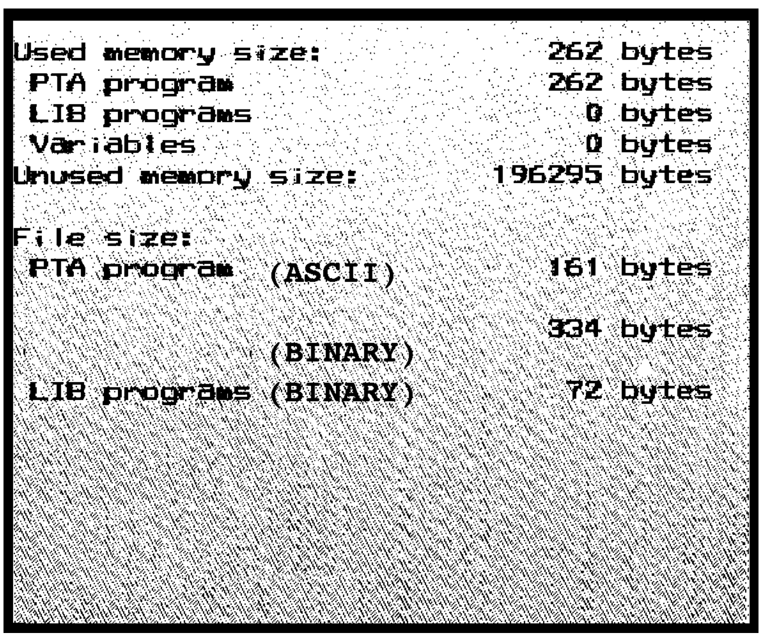
(1) Function

This command displays on the screen the used memory size of the program area in which a PTA program/library is stored and the memory size required to store to a memory card.

(2) Format

```
PMEMO
```

(3) Output example



Total size of used memories of program area

Not used

Memory size required to store to memory card

Immediate Execution Command

(1) Function

When a statement with no line number is input and the ⏎ (RETURN) key is pressed, the statement is immediately executed.

However, GOTO, GOSUB, RETURN, RETMAIN, IF, FOR, NEXT DATA, RDATA, RESTORE and CHAIN, CALLIB statements are not immediate execution commands.
See Section 4 for these statements.

(2) Format

```
Statement
```

RUN Command

(1) Function

This command starts PTA program/library execution. Execution is terminated when the STOP statement is executed, when an error occurred, or when the [RESET] key is pressed.

(2) Format

```
[RUN] key or
RUN [[operand 1][,operand 2]]
```

operand 1: Start-line number
operand 2: Suspended-line number

1) `RUN`	Starts execution from statement on first line
2) `RUN 1ØØ`	Starts execution from statement on line 100
3) `RUN ,5ØØ`	Starts execution from statement on first line, and suspends execution on line 500
4) `RUN 1ØØ,5ØØ`	Starts execution from statement on line 100, and suspends execution on line 500

Note: Contents of variables are not initialized by the RUN command.

STOP Command

(1) Function

This command stops the PTA program/library in execution.

(2) Format

```
[STOP] key
```

CONT Command

(1) Function

This command resumes the suspended program execution.

Note that this command can only be executed when program execution is suspended after execution of the RUN or STEP command.

(2) Format

```
[CONT] key
CONT [operand]
```

1) `CONT`	Restarts program from next on suspended line.
2) `CONT 1ØØØ`	Restarts program from next on suspended line, and suspends execution on line 1000.

RESET Command

(1) Function

This command stops command or PTA program/libraries execution.

(2) Format

```
[RESET] key
```

(3) Initialization

This Command : 1. Clears system variables EX1, EX2, EX3, EX4, and EX5.
2. Clears user-defined variables. Common variables are not cleared.

SAVE Command

(1) Function

This command saves a PTA program to a memory card. In this case, the file size of the PTA program must be smaller than the unused memory size of the memory card.
The file size of the PTA program and the unused memory size of the memory card are output on the screen by executing the PMEMO command and the PLIST command, respectively.

(2) Format

```
SAVE PTA program name [.Attribute][,operand 1][,operand 2]
```

- PTA program name: Alphanumeric string up to 6 characters starting with an uppercase alphabetic character.
- .Attribute: .PTA or .IMG
- operand 1: Start-line number
- operand 2: End-line number

Notes:

- The file opened by CALL OPNI (or OPNO) "% file name" is closed when this command is executed.
- Labels can be used as operands 1 and 2.
- Before saving a program, make sure the memory card is formatted. When saving to an unused memory card, format the memory card in advance.
  For formatting mathod of the memory card, refer to paragraph 4.5.2 of Panel Operation Part in the Operation Manual.
- When .PTA is specified as attribute, the program is saved as an ASCII file. When .IMG is specified, the program is saved as a binary file, which has a shorter loading time. As the default attribute, .PTA is automatically selected for saving.

LOAD Command

(1) Function

This command loads a PTA program loaded on a memory card and stores it to the program area in the main frame. All the PTA programs already stored in the user program area are replaced by the new program unless OVERLY is executed.

(2) Format

```
LOAD PTA program name [.Attribute]
```

Alphanumeric string up to 6 characters starting with an uppercase alphabetic character. .PTA or .IMG

Notes:

- The file (opened by CALL OPNI (or OPNO) "% file name") is closed when this command is executed.
- When reset during program loading, part of the programs is loaded.
- The spectrum analyzer program area (memory) is backed up by a battery. Therefore, the program contents are not lost even when the power switch is turned off.

OVERLAY Command

(1) Function

This command specifies to overwrite the current PTA program during LOAD command execution.

(2) Format

```
OVERLAY
```

Note: This state continues until the RESET command is executed.

PDEL Command

(1) Function

This command deletes the PTA programs stored in a memory card.

(2) Format

```
PDEL PTA program name or PTA library file name [.Attribute]
                                                  ↑
                                          PTA, IMG, LIB, LIA
```

Notes:

- "% file name" (data files) cannot be erased by the PDEL command.
- The file (opened by CALL OPNI (or OPNO) "% file name") is closed when this command is executed.
- When attribute is omitted, .PTA is automatically selected as the default attribute for saving.

PLIST Command

(1) Function

This command displays on the CRT screen the names and sizes of files stored on memory card along with the amount of unused memory.

(2) Format

```
[PLIST] key
```

(3) Output

This command causes the screen to scroll by page (24 lines) unit.
When more than 17 files are stored on a memory card, the files cannot be displayed on one page, therefore a screen such as 1) below is displayed. The screen is displayed page by page by using the PLIST command repeatedly. When the contents can be displayed on a single page, a screen such as 2) is displayed.

1) When pages follow

…………	……bytes	PROG (IMAGE)
%SDAT0.DAT	1024 bytes	DATA
%SDAT2.DAT	1024 bytes	DATA
ABCXYZ.PTA	15808 bytes	PRJG (ASCII)
		continue

2) When no pages follow

BANDLH.PTA	18568 bytes	PROG (ASCII)
RPLLH.IMG	35786 bytes	PRJG (IMAGE)
MAXMIN.LIB	27368 bytes	LIBRARY

unused memory size : 89010 bytes

Unused memory size : Indicates unused memory size (No. of bytes) of the memory card.

NOTES

- The file (opened by CALL OPNI (or OPNO) "% file name") is closed when this command is executed.
- Only the PTA program file, PTA library file and data file created by the PTA are displayed by the PLIST command. Therefore, since the spectrum analyzer does not display the saved waveform and measurement parameters, if they exist, the unused memory size is reduced.

STARTP Command

(1) Function

Turns on the PTA and registers the start-up function, which loads and executes the specified PTA program when the power is turned on.
This function can be separately registered and set for a PTA program on a memory card and a PTA program in the main frame.

(2) Format

```
STARTP program name
STARTP @
```

`STARTP program name`	: Register for PTA program on memory card
`STARTP @`	: Register for `spectrum analyzer` internal PTA program

1) Start-up function registration for PTA program on memory card
   - When the power is turned on after this function is registered, the PTA is turned on and the registered PTA program is loaded and executed.
   - When this function is registered, a special "p2110. bat" file is created on the memory card. (This file is not displayed by the PLIST command.)
   - In the following cases, the start-up function is not performed even if registered:
     - When a memory card is not inserted when the power is turned on.
     - When a PTA program with the registered program name is not found on the memory card.
     - If the power was turned on while pressing the [PTA : 7] key.
   - This function is executed first even if start-up function is registered for the internal program of the main frame.
   - When start-up function is executed, the PTA program is loaded from the memory card, and the previous program in the main frame is cleared. Also, when start-up function is registered for the internal PTA program, it is cleared too.
   - If both "STARTP" and "STARTP@" are registered, the file registered by the STARTP command is executed preferentially.

2) Start-up function registration for spectrum analyzer internal PTA program
   - When the power is turned on after this function is registered, the PTA is turned on and the spectrum analyzer battery back-up PTA program is run automatically.
   - When there is no PTA program in the spectrum analyzer, this function cannot be registered.
   - The start-up function is not performed in the following cases:
     - When the memory card start-up function was executed first.
     - When a new PTA program was loaded after the start-up function was registered. (In this case, start-up function registration is canceled.)
     - When there is no PTA program in the spectrum analyzer.
     - If the power was turned on while pressing the [PTA : 7] key.

CANCEL Command

(1) Function

Cancels start-up function registration.

(2) Format

`CANCEL` : Register for PTA program on memory card
`CANCEL @` : Cancel registration for `spectrum analyzer` internal PTA program

- When start-up registration for memory card is canceled, the "p2110. bat" file is deleted.
- When the power is turned on while pressing the [PTA : 7] key, the start-up function is temporarily canceled, but the function registration status does not change.

EDITLIB Command

(1) Function

This command defines a new PTA library, or specifies a PTA library as the object of the program execution and program edition commands.

(2) Format

```
EDITLIB [PTA library name]
```

Alphanumeric string with up to 8 characters starting with a capital alphabet

Characters available for the 2nd character on :

Under bar	
Capital alphabet	: A to Z
Small alphabet	: a to z
Numeral	: 0 to 9

However, small alphabets are converted to capitals.

- When the EDITLIB command is executed specifying the name of a new PTA library as a parameter, the registration of the specified PTA library is started. The PTA library can be registered by inputting a statement with a line No.
- When the EDITLIB command is executed specifying the name of an already registered PTA library as a parameter, a library program to be the object of program execution and edition commands is specified.
- When the EDITLIB command is executed without a parameter, the name of the currently specified library is displayed.
- The PTA library name specified by the EDITLIB command is displayed at the bottom right of the screen.

EDITPTA Command

(1) Function

This command specifies PTA programs as the object of edition and execution.

(2) Format

```
EDITPTA
```

- Select PTA programs as the object of edition and execution. The object of processing is switched to PTA programs by executing the EDITPTA command during PTA library selection. Additionally, immediately after PTA ON, always PTA programs are selected.

RENAME Command

(1) Function

This commands changes the name of the specified PTA library.

(2) Format

```
RENAME PTA library name,PTA library name
```

New PTA library name

Program name to be changed
Alphanumeric string with up to 8 characters starting with a capital alphabet
Characters available for the 2nd character on :

Under bar	
Capital alphabet	: A to Z
Small alphabet	: a to z
Numeral	: 0 to 9

However, small alphabets are converted to capitals.

- The name of an already registered PTA library is changed. It is not allowed to specify the already registered PTA library name for the new PTA library name.

LIBMEM Command

(1) Function

This command displays a list of PTA libraries in the memory.

(2) Format

```
LIBMEM
```

- Names of library programs in the memory are displayed in list form. If the list cannot be displayed at a time, re-execute the LIBMEM command to display the next page. If there is no library in the memory, nothing is displayed.

SAVELIB Command

(1) Function

This command saves the specified measuring instrument library program to a memory card with the specified file name.

(2) Format

```
SAVELIB File name [,PTA library name...]
```

PTA library name: Alphanumeric string with up to 8 characters starting with a capital alphabet

Characters available for the 2nd character on :

Under bar
Capital alphabet : A to Z
Small alphabet : a to z
Numeral : 0 to 9

However, small alphabets are converted to capitals. Library names can be specified up to ten names by separating them with commas (,). If no name is specified, all the PTA libraries residing in the memory are specified.

File name: Alphanumeric string with up to 6 characters starting with a capital alphabet Characters available for the 2nd character on :

Under bar
Capital alphabet : A to Z
Small alphabet : a to z
Numeral : 0 to 9

However, small alphabets are converted to capitals.

- The PTA library is saved in intermediate code form. The file extender is ".LIB".

SECTION 4

PTL

TABLE OF CONTENTS

SECTION 4
PTL

PTL (Personal Test Language) is a programming language similar to BASIC.
It consists of basic PTL statements and extended PTL (including system variables, system subroutines, and GPIB statements).

Elements of Statement Configuration

Line number

(1) Function

A line number is placed at the beginning of each statement and serves as an index during program editing or execution.

(2) Format

```
Numeric String
```

Integer constant from 1 to 65535

Constants

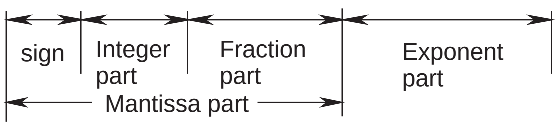
(1) Function

A constant represents a specific numeric value, character string or bit string.

(2) Format

(a) Numeric constants

```
[—]numeric string [.numeric string][E[—]numeric string]
```



The maximum number of mantissa digits is 15(including a sign and a decimal point.) and the range of exponent part is 10^{308} to10^{-307} .

When a numeri c constant is assigned to an integer type numeric variable, the range is —32768 to +32767.

(b) Character constants

```
"String"
```

1 to 255 characters enclosed with double quotation marks (" ")

Note: One line of program corresponds two lines on screen. Then, maximum number of characters on a program line is limited to the value.

(c) Bit constants

- Hexadecimal constant

  $ <u>Hexadecimal expression</u>

  0 to FF

- Binary constant

  # <u>Binary expression</u>

  0 to 11111111

(3) Examples

(a) Numeric constants

```
1
–12.3
12E3                 …Equal to 12000
–Ø.12E–3             …Equal to –0.00012
```

(b) Character constant

```
"Who are you? "
```

(c) Bit constants

```
$F                   …Equal to #1111 (binary) or 15 (decimal).
#ØØØ11Ø1Ø            …Equal to $1A (hexadecimal) or 26 (decimal)
```

Variables

Variables include local, common and system variables. For the system variable, see Section 5, "System Variables".

(1) Local variables

A local variable is one that is effective in a PTA program/library only.
Local variables include simple and array variables.

- Simple variable

There are numeric, character string, and bit string variables. The simple variable consists of eight or less characters, the first of which must be an upper-case alphanumeric character as shown below:

- Real number-type numeric-variable name: Upper-case alphabetic character [alphanumeric [alphanumeric]] —— ABCD0123
- Integer-type numeric-variable name: Upper-case alphabetic character [alphanumeric [alphanumeric]] % —— A%
- Character-string-variable name: Upper-case alphabetic character [alphanumeric [alphanumeric]] $ —— ABC$
- Bit-string-variable name: Upper-case alphabetic character [alphanumeric [alphanumeric]] # —— A#

- Array variable

The variable (declared as an array by the DIM statement) is called an array variable. Some system variables are also handled as array variables. The format of the array variables is shown below.

- Array variable : variable (numaric constant or numeric variable)

Subscropt range
- For numeric-array variable 0 to 1023
- For character-string-array variable 0 to 254
- For bit-string-array variable 0 to 7

NOTES

- The subscript range for an array variable is from 0 to array size –1.
- When the subscript in the array variable is a real number, it is truncated after the decimal point.
- Up to 256 variables can be used (except for system variables).
- Pre-registered symbols (such as commands, statements, functions and system variables) cannot be used as user-defined variable names.

(2) Common variables

Common variables are ones that can be commonly accessed from all programs (PTA program/library). The name of a common variable starts with "@" followed by capital alphabets. The length of a common variable name is 8 characters at longest, including the @ mark.
Values of common variables are retained until the RESET command or COMCLEAR command is executed. Common variables include simple variables and array variables:

- Simple variables

There are numeric, character string and bit string variables.

- Real number variable name: @ + variable name
- Integer numeric variable name: @ + variable name + %
- Character string variable name: @ + variable name + $
- Bit string variable name: @ + variable name + #

- Array variables

Like array local variables, array common variables are declared by a DIM statement.

The DIM statement may be declared in any of programs, and double definition is also allowed. The array size is linear or quadratic.

- Real number variable name: @ + variable name (array size [, array size])
- Integer numeric variable name: @ + variable name + % (array size [, array size])
- Character string variable name: @[alphanumerics[alphanumerics]]$ (array size [, array size])
- Bit string variable name: @ @[alphanumerics[alphanumerics]]# (array size [, array size])

Multi statement

By using ' & ' as the delimiter in a statement, multiple statements can be entered on the same line.
This delimiter can also be used to enter a program of two lines. There are no restrictions on the number of statements within a program, provided that the length of the program does not exceed two lines.

Example :
```
1Ø FOR I=Ø TO 1Ø & A=I*I & PRINT A & NEXT I
2Ø STOP
```

Functions

There are basic functions (arithmetic, boolean, statistical and character-string functions) and dedicated functions in PTL. The system functions are used for measurement evaluation.

(1) Arithmetic function

Function name	Function	Parameter	
Sine	`SIN(X)`	The X unit is degrees.	A constant or a variable os used for X.
Cosine	`COS(X)`		
Tangent	`TAN(X)`	$X \neq \pm 90(2n+1)$, n:any integer	
Arcsine	`ASN(X)`	$\lvert X \rvert \leq 1$	
Arccosine	`ACS(X)`		
Arctangent	`ATN(X)`		
Natural logarithm	`LN(X)`	$X > 0$	
Common logarithm	`LOG(X)`		
Exponent	`EXP(X)`		
Square root	`SQR(X)`	$X \geq 0$	
Absolute value	`ABS(X)`		
Sign	`SGN(X)`	FOR X > 0 , SGN(X) = 1 FOR X < 0 , SGN(X) = –1 FOR X = 0 , SGN(X) = 0	
Integer value	`INT(X)`	X : Numeric type constant variable (An integer less than X is returned.)	
Rounding up	`ROUND(X[,N])`	X : Numeric type constant variable N : Numeric type constant variable (default value: N = 0) (X is rounded up to the N-th decimal place.)	
Function to calculate the quotient and remainder	`Q=DIV(R,S,D)`	Q : Numeric variable ------ Stores the quotient R : Numeric variable ------ Stores the remainder S : Numeric variable ------ Stores the dividend D : Numeric variable ------ Stores the divisor	
Function to isolate the integer and decimal parts of a real number	`I=FIX(S,D)`	I : Integer variable ------- Stores only the integer part S : Real-number variable --- Stores the real number of the original value D : Real-number variable --- Stores only the decimal part	

(2) Boolean functions

Function name	Function	Parameter
Negation	`NOT(X)`	X and Y are constants and variable of bit type or numeric type, and hexadecimal constants.
Logical product	`AND(X,Y)`	
Logical sum	`OR(X,Y)`	
Exclusive OR	`EOR(X,Y)`	

(3) Statistical functions

Function name	Function	Parameter
Function to find maximum value	`MX=max(S)`	S : Variable defined as one-dimensional arrey MX : Stores the maximum value MN : Stores the minimum value SM : Stores the sum total MS : Stores the mean value VR : Stores the variance Variance = $\frac{\Sigma (X - \bar{X})^2}{\text{No of samples}}$
Function to find minimum value	`MN=min(S)`	
Function to find sum	`SM=sum(S)`	
Function to find mean value	`MS=mean(S)`	
Function to find variance value	`VR=var(S)`	
Function to find all above values	`VR=sta (S,MX, MN,SM,MS)`	

NOTES

The left side always consists of numeric variable in which found (calculated) value is stored.
The one-dimensional S-parameter is valid even if there is only one element provided.
When all the elements are to be processed statistically, no subscript is necessary at the entry.
If a subscript is included, only the element specified by the subscript will be processed.

(4) Character-string functions

(a) Interchange between numerics and characters (strings)

1. ASC (Alphabetic constant or variable)

   ASC generates the character code for the first character of the string.

2. CHR$ (Constant or variable)

   CHR$ generates the character with the character code corresponding to the parameter value.
   For a character type, the character remains unchanged. The parameter range is from 0 to 255.

3. STRING$ (Numeric constant or variable, constant or variable, character constant or variable)

   STRING$ generates the characters (with the character code of the numeric value or the first character of string specified by the 2nd parameter) by the number of characters specified by the 1st parameters.Up to 255 repetitions may be specified.
   Refer to CHR$ ( )

4. HEX$ (numeric-value-type constant or variable 1 [, numeric-value-type constant or variable 2])

   A decimal value of the first parameter is given as a hexadecimal character string with number of digits specified by the 2nd parameter.
   An error will occur if the value of the first parameter does not fall in between -2^{31} and $2^{32}-1$. An error will occur if the second parameter goes beyond eight digits. When omitted, the return value will be of variable length.

5. OCT$ (Constant or variable)

   OCT$ generates the octal character string corresponding to the parameter value. An error is generated when the range –32768 to 32767 is exceeded.

6. BIN$ (numeric-value-type constant or variable l [, numeric-value-type constant or variable 2])

   A decimal value of the first parameter is given as a binary character string with number of digits specified by the 2nd parameter.
   An error will occur if the value of the first parameter does not fall in between -2^{31} and $2^{32}-1$. An error will occur if the second parameter goes beyond 32 digits. When omitted, the return value will be of variable length.

7. CVI (Character constant or variable of 2 or more characters)

   CVI generates the value converted from a character string to an integer numeric expression. If the character string exceeds two characters, the excess part is disregarded. Conversely, an error is generated when it is less than 2 characters.

8. CVD (Character constant or variable of 8 or more characters)

   CVD generates the value converted from a character string to a double-precision real-number numeric expression. When the character string exceeds 8 characters, the excess part is disregarded. Conversely, an error is generated when it is less than 8 characters.

9. MKI$ (Integer constant or variable)

   MKI$ generates the corresponding character code of the internal binary expression of the specified numeric value.
   This is the reverse process of the previously-mentioned CVI.

10. MKD$ (Double-precision real-number constant or variable)
    MKD$ generates the corresponding character code of the internal binary expression of the specified numeric value. This is the reverse process of the previously-mentioned CVD.

11. VAL (Character variable, Number constant or variable 1, numeric constant or variable 2)
    VAL isolates the mth to nth numeric characters (including other than numeric code) of the specified data string and changes them to the double-precision real-number numeric expression, assuming that m and n are the specified values by variable 1 and variable 2, respectively.
    Both m and n may be omitted. When m is omitted, the object runs from the head character of the data string: and when n is omitted, the object runs to the last character of the data string.
    An error occurs when no numeric character is found.

12. BVAL (character constant or variable)
    This function will convert the parameter string notated in binary into an unsigned decimal value.
    An error will occur if the parameter exceeds 32 bits. All characters other than “0” or “1” will be ignored.

13. HVAL (character constant or variable)
    This function will convert the parameter string notated in hexadecimal into an unsigned decimal value.
    An error will occur if the parameter exceeds 32 bits (8 characters). Characters other than “0” to “9” and “A” to “F” are ignored.

14. CHR (Numeric constant or variable)
    CHR generates the same character string as that to be displayed by the PRINT statement within the specified numeric value by parameter.

15. STR$ (Numeric constant or variable)
    This performs exactly the same processing as described for the CHR function.

(b) Retrieving character strings

1. INSTR ([Numeric constant or variable,] character constant or variable 1, character constant or variable 2)
   When character string 2 is found within character string 1, its position is returned; if it is not found, 0 is returned. When the numeric value is included in the 1st parameter, the search starts from the indicated position with the numeric value; when it is omitted, the search starts from the header. The range of the value is from 1 to 255.

2. LEFT$ (Character constant or variable, numeric constant or variable)
   This gives the specified number of characters (counting from the left) as specified by the second-parameter. When the specified number exceeds the number of characters in the strings, whole the character string is given. The specifiable number is from 0 to 225. When the specified number is 0 , a null string is returned.

3. MID$ (Character constant or variable, numeric constant or variable 1, numeric constant or variable 2)
   This gives the n of character strings from the m-th character, assuming that the m and n are the specified values by the variable 1 and variable 2, respectively. The range of m/n is (1 to 256) / (1 to 255), respectively. When m exceeds the total number of characters, a null string is returned.

4. RIGHT$ (Character constant or variable, numeric constant or variable)
   This performs the same processing as the LEFT$ ( ) command but from the right side. The value range is also the same (0 to 255). Note that this command does not reverse the character string sequence.
5. LEN (Character constant or variable)
   LEN gives the number of characters in a character string including all character codes from 0 to $1F.
6. SLEN (character type constant or variable)
   This gives the number of characters composing a character string in the same manner as specifying a value in LEN ( ). However, this gives the length with the space at the end of the character string omitted .
7. SGET$ (character type constant or variable)
   This gives a valid character string with the space at the end omitted.

(5) Dedicated functions

Function description	Function	Parameter
Reads the error code and line number in which error occurred on	`V=ERRREAD(m)`	m 0 : Error code 1 : Line number in which error occurred
Reads the type of event	`A#=STATUS(m)`	m 0 : Event 0 1 : Event 1 2 : Event 2 3 : Event 3
Reads the date and o'clock, minute, second	`A$=DTREAD$(m)`	m 0 : Date (YY-MM-DD) 1 : o'clock, minute, second (HH:MM:SS)
Random number generation (more than 0, less than 1)	`RND(m)`	m : Specify an arbitrary value.

NOTES

- ERRREAD (m) can only be used during at error interrupt. For details on error interrupts, see Section 4, "ON ERROR statement".
- STATUS (m) can only be used during an event interrupt. For details on event interrupts, see Section 4, "ENABLE EVENT statement".
- m is a numeric constant or numeric variable.
- The sequence of pseudo-random numbers generated by RND(m) becomes the same each time RUN is executed.
  See Section 4, "RNDMIZE statement" for how to change the sequence.

Arithmetic operators

(1) Function

These operators perform addition, subtraction, multiplication, division, and exponential operations.

(2) Format

= … Substitution
+ … Addition
– … Subtraction
* … Multiplication
/ … Division
! … Exponentiation
( ) … Represents operation priority
(Operations in parentheses are performed first.)

(3) Operation Priority

The operation priority is shown below.

Table 4-1 Operation priority of arithmetic operators

Operation priority	Arithmetic operators
High	!
↕	* /
	+ -
Low	=

NOTES

- Bits and characters cannot be used in operations.
- If X of X ! Y is a minus number, but Y is a plus number, X ! Y can be operated.
- If there is a different type variable on the right side of an equals sign (=), an overflow or underflow error may occur.
- Number of digits of divided becomes number of digits of the solution on division with numerals or variables.

(4) Example

```
A$="abc"
C=(D+1ØØ)/E
J=((K+1)*1Ø–M)*1Ø
```

Relational operators

(1) Function

These operators perform relational operations.

(2) Format

`=`	…	Equal (=)
`>< or <>`	…	Not equal (≠)
`>`	…	Greater than (>)
`<= or =<`	…	Equal to or less than (≤)
`<`	…	Less than (<)
`>= or =>`	…	Equal to or greater than (≥)

(3) Comparing character strings

When comparing the sizes of character strings, count only significant characters.
(Ignore any spaces at the ends of the character strings to the left and right of an operator)

- If two character strings are the same length, their characters are compared sequentially from the beginning. The first character which is different is found. The character which has the lower code value will determine the smaller character string.

  Example : ABC is smaller than ABX.

- If two character strings are different lengths, the character strings over their common length are compared. If the two strings are equal over this length, the shorter character string will be the smaller character string.

  Examples : ABX is larger than ABCD.
  ABC is smaller than ABCD.

- The smallest character string is one with 0 length.

  Example : The length of A$ is 0 when DIM A# (10) is declared.

(4) Examples

```
IF C=Ø GOTO 1ØØ

IF JKL>=168 STOP
```

String concatenation (the "+" operator)

(1) Function

String concatenation is possible with the "+" operator.

(2) Format

```
{ character string constant }   { character string constant }
{ character string variable } + { character string variable }
{ character string function }   { character string function }
```

Notes:

- Only be used with the right hand parameter of the LET statement.
- You cannot concatenate character string and numeric values, character string and bit, or bit and bit.

(3) Examples

```
1ØØ  A$="ABC"
11Ø  B$="DEF
12Ø  A=INSTR(A$,"␣")–1
13Ø  B=INSTR(B$,"␣")–1
14Ø  C$=LEFT$(A$,A)+LEFT$(B$,B)
15Ø  PRINT "A$=",A$
16Ø  PRINT "B$=",B$
17Ø  PRINT "C$=",C$

AS=ABC␣␣␣␣␣␣␣␣
B$=DEF␣␣␣␣␣␣␣␣
C$=ABCDEF␣␣␣␣␣
```

Space

NOTES

- Simple character-string variables are assumed to be a ten-character array-declared variables, implicitly. Therefore, characters not assigned will be filled with spaces. For details, see Section 4, "Display (PRINT statement)" and "Reverse display (PRINTR statement)".
- By using the above method, you can concatenate actual stored character only.

Formats

(1) Function

These formats specify the format of strings in output operations. Integers, real numbers without exponents, real number with exponents, strings, binary numbers, and hexadecimal numbers can be specified.

(2) Formats

- Integer
  : I number of digits
  (1 to 18)
- Real number without exponent
  : F number of all digits. number of fractional digits
  (4 to 20)
  (Number of all digits ≥ number of fractional digits+3)
- Real number with exponents
  : E number of all digits. number of fractional digits
  (9 to 24)
  (Number of all digits ≥ number of fractional digits+8)
- String
  : C number of digits
  (0 to 255)
- Binary number
  : B number of digits
  (1 to 8)
- Hexadecimal number
  : H number of digits
  (1 or 2)

(3) Examples

```
PRINT A$:C3,J:F1Ø.4
```

NOTES

- When number of digits is 0 for string, the character length becomes variable to output all actual length of the character string variable.
- A single space is included at the end of each PRINT statement provided that the FORMAT specifiers are capitalized. These spaces can be omitted by using a small-case FORMAT specifier instead of a capitalized FORMAT specifier (See Section 4, "Display(PRINT statement)" and "Reverse display (PRINTR statement)".)

Label

(1) Function

A jump address can be assigned indirectly by using a label with a line number in a statement such as GOTO or GOSUB.

(2) Format

```
Line number␣∗ label
Line number␣∗ label␣statement
```

- A label consists of up to eight alphanumeric characters starting with an uppercase alphabetic character. The label is prefixed with ∗.
- When multiple line numbers are defined with the same label, an error occurs during program execution.

(3) Examples

```
 1Ø   INPUT A
 2Ø   IF A=Ø GOSUB ∗ ABC1
 3Ø   IF A<>Ø GOSUB ∗ ABC2
 4Ø   GOTO 1Ø
1ØØ   ∗ ABC1
11Ø   PRINT "OK!"
12Ø   RETURN
2ØØ   ∗ ABC2
21Ø   PRINT "NG!"
22Ø   RETURN
```

Basic Statements

Comment (REM statement)

(1) Function

This statement gives comments to program. These comments are not executed by the system and they have no effect on program execution.

Note: When a specific statement is described as a comment statement, it must be enclosed by a pair of double quotation marks(" ") as a character constant.

(2) Format

```
REM ["comment"] or
' [comment]
```

(3) Examples

```
1Ø  REM
2Ø  REM "Compute average"
3Ø  'Compute average
4Ø  A=1ØØ 'Initial set
```

Array declaration (DIM statement)

(1) Function

This statement declares arrays. Arrays must be one-dimensional or two-dimensional, and are restricted at a size as shown in paragraph (2) below according to the type of variable name.

(2) Format

```
DIM variable-name(array-size[,array-size])
    [,variable-name(array-size[,array-size])....]
```

Notes:

- The same variable name cannot be redefined as an array. A variable (that has been used as an independent variable) cannot be declared as an array.
- Error W225 will be generated when a two-dimensional array is referred to without the specification of two dimensions.
- Error W224 will be generated when a one-dimensional array is referred to as a two-dimensional array.
- The size limit of the declarable array is as follows. If the declared size exceeds these limits, ERROR 203 will be generated.

Character type ...... 1 to 255	Two dimensional array:		
Bit type .... 1 to 8	One dimensional side	Two dimensional side	
Numeric type .... 1 to 1024	1 to 1024	Character type	1 to 255
		Bit type	1 to 8
		Numeric type	1 to 1024

- For the numeric type, the program area will become insufficient; thus, it is impossible to define 1024 on both the one- and two-dimensional sides. In this case, ERROR 206 will be generated.
  The total number of array elements that can be declared (product of the number of one-dimensional array elements by the number of two-dimensional array elements) is not restricted because it depends on the capacity of empty memory.
- For the character array, ten characters long are automatically declared when no array is declared.
- For the bit type, array eight bits long are automatically declared when no array is declared.
- Error W224 occurs when individual elements are referred to (read or written) without the appropriate array declaration.

(3) Examples

```
DIM  CARR(1ØØ),A$(5,12)

DIM  I#(8),ALP$(4Ø)
```

(4) System variables which have been unconditionally declared as arrays.

```
XMA(∗),XMB(∗),
XMT(∗),XMB(∗),SMA(∗),SMB(∗),SMT(∗),IMA(∗),IMB(∗),RMA(∗),RMB(∗)
```

NOTES

∗ is an array element of 0 to 500.

Initialization (CLEAR statement)

(1) Function

Initializes user-defined variables.

(2) Format

```
CLEAR
```

Note: When the CLEAR statement is executed, the array can be redefined since variables are re-initialized in a manner similar to that in which executing RESET is executed.

Substitution (LET statement)

(1) Function

This statement substitutes variables for constants, variables, and results of operations.
See Section 4, "Arithmetic operators".

(2) Format

$$\texttt{[LET]variable=[(]}\left\{\begin{array}{l}\texttt{constant}\\\texttt{variable}\\\texttt{function}\end{array}\right\}\texttt{[)]}$$

$$\texttt{[arithmetic operator[(]}\left\{\begin{array}{l}\texttt{constant}\\\texttt{variable}\\\texttt{function}\end{array}\right\}\texttt{[)] ...]}$$

$$\begin{array}{c}\texttt{+ -}\\\texttt{* /}\\\texttt{!}\end{array}$$

$$\texttt{[LET]character type variable=}\left\{\begin{array}{l}\texttt{character string constant}\\\texttt{character type variable}\\\texttt{character string function}\end{array}\right\}\texttt{+}$$

$$\left\{\begin{array}{l}\texttt{character string constant}\\\texttt{character type variable}\\\texttt{character string function}\end{array}\right\}\texttt{+ ...}$$

Notes:

- Bits and characters cannot be used in operations.
- If a substitution statement is placed after an IF statement, LET cannot be omitted.

(3) Examples

```
LET A=B+C or A=B+C

IF X=Ø LET Y=1Ø
```

Branch (GOTO statement)

(1) Function

This statement changes the sequence of program execution to the statement of the specified line number.

(2) Format

```
GOTO line number or GOTO *label
```

Termination of execution (STOP statement)

(1) Function

This statement terminates program execution after displaying an execution termination message on the CRT screen as follows.

```
STOP IN  line  number
```

(2) Format

```
STOP
```

Note: Suspension specifications are ignored in STOP statements, since program execution is terminated.

Branch to subroutines (GOSUB statement)

(1) Function

This statement changes the program execution to the subroutine with the specified line number. When the RETURN statement is executed at the end of the subroutine, the program execution is returned to the statement following the GOSUB statement.

(2) Format

```
GOSUB line number or GOSUB *label
```

Note: Calling another subroutine during execution of a subroutine is referred to as "nesting". Up to 10 nesting levels are permitted.

Return from subroutines to main routine (RETMAIN statement)

(1) Function

When the RETMAIN command is used during program execution, control is returned to the highest level of the routine regardless of the nesting level.

(2) Format

```
RETMAIN
```

Note: If the RETMAIN command has been executed in the highest level of the routine, ERROR F213 occurs.

Return from subroutines (RETURN statement)

(1) Function

This statement returns program execution from the subroutine to the statement following the corresponding GOSUB statement.

(2) Format

```
RETURN
```

Decision (IF statement)

(1) Function

If the result of the relational operation is true, this statement executes the subordinate statement. For relational operators, see Section 4, "Relational operators".

(2) Format

IF $\left\{\begin{array}{l}\text{constant}\\ \text{variable}\end{array}\right\}$ relational operator $\left\{\begin{array}{l}\text{constant}\\ \text{variable}\end{array}\right\}$ statement

=
>< or <>
>
<= or =<
<
>= or =>

Notes:

- All statements including IF statements can be placed as subordinate statements.
- Relational operations can not be performed among numerical values, characters, and bits.
- If a substitution statement is placed after an IF statement, LET cannot be omitted.

(3) Examples

```
IF C=1 GOTO 1ØØ

IF ACH$=BCH$ PRINT ACH

IF C<1Ø IF C>=2Ø PRINT "ERROR"

IF C<1Ø LET C=1Ø
```

Repetitions start (FOR statement)

(1) Function

This program loop causes the program code (located between the FOR and NEXT) to be repeatedly executed, until the specified variable is equal to or greater than the specified end value.

Up to 10 nesting levels may occur within a FOR statement.

(2) Format

$$\texttt{FOR numeric variable =}\underset{\text{Initial value}}{\left\{\begin{array}{l}\texttt{numeric constant}\\ \texttt{numeric variable}\end{array}\right\}}\ \texttt{TO}\ \underset{\text{Ending value}}{\left\{\begin{array}{l}\texttt{numeric constant}\\ \texttt{numeric variable}\end{array}\right\}}$$

$$\texttt{[STEP}\underset{\text{Increment (default value is 1)}}{\left\{\begin{array}{l}\texttt{numeric constant}\\ \texttt{numeric variable}\end{array}\right\}}\texttt{]}$$

Notes:

- Even if the initial value exceeds the end value, one operation cycle will be performed.
- NEXT statements may be used anywhere; however, for proper execution they must be properly positioned.

(3) Example

```
FOR C=1 TO 1ØØ
FOR T=TB TO TE STEP Ø.1
FOR D=–1 TO –1Ø STEP –1
NEXT D
NEXT T
NEXT C
```

Repeats

Repetition termination (NEXT statement)

(1) Function

This statement is used with its corresponding FOR statement to terminate the repeated operation.

(2) Format

```
NEXT numeric variable
```

Same variable as that specified in FOR statement

Key-input (INPUT statement)

(1) Function

This statement is used to assign data input from the front panel key to variables.
When the statement is executed, the following message is displayed on the CRT.

? □

Input data after the display question mark ? via the numeric key of the front panel, then press [ENTER] key of the instrument.

Use commas (,) as delimiters of data if required.

(2) Format

```
INPUT ["displayed character string",] variable[,variable....]
```

Notes:

- If a real number is input for an integer variable, it is truncated under decimal point.
- If the input data length is smaller than that which has been declared, spaces are appended to the entry. If it is greater, the excess digits will be truncated.
- For numeric and bit type variables, spaces before and after the input value are ignored.
- Hexadecimal data cannot be input.
- Five variables can be specified .
- The ,(comma) and –(minus) are input by pressing the [kHz] key and the [MHz] key of the front panel, respectively.

(3) Examples

```
INPUT "COUNT=",C  → COUNT=? 123
INPUT C,A$,I#     → ? 123,Q,1Ø11Ø1
```

Display (PRINT statement)

(1) Function

This statement edits and displays data on the CRT screen.
Unformatted data is displayed with spaces added after its effective digits. The format name and output formats are shown in Table 4-2.
For the format, see Section 4, "Formats".
Line feed is disabled by adding " ; " at the end.

Table 4-2 Format Name and Output Format

Format name	Output format
`I`	Zero-suppressed integer (Ex. ␣␣ 123)
`F`	Zero-suppressed integer and zero-suppressed fraction (Code digit exists.) (Ex. ␣␣123.45␣)
`FP`	Zero-suppressed integer and zero-suppressed decimal number (unsigned) (Ex. ␣123.45␣)
`E`	{ ␣ / – } Zero-suppressed fraction E [–] exponent (Ex. ␣1.23E–2␣)
`C`	String … If the size of data is smaller than the specified format size, spaces are added; and if it is greater, the excess lower digits are truncated.
`B`/`H`	Zero-suppressed binary-number/hexadecimal-number string (Ex. ␣␣1011)

(2) Format

```
PRINT {variable [:format]} [, {variable [:format]} …][;]
      {string constant   }    {string constant   }
```

Constant displayed as is

No line feed

Notes:

- Up to five variables or constants can be specified.
- Values which cannot be expressed are displayed as ***...*.
- A string-which is an array of character variables- is comprised as follows:

String data A$="A B C D E"

Array variable

A(4)
A(3)
A(2)
A(1)
A(0)

- A binary numeric variable- which is an array of binary digits- is comprised as follows:

Binary numeric dara 1#=#1 0 0 1 1

I(0)
I(1)
I(2)
I(3)
I(4)

- The last space can be deleted by using a lower-case format i, f, fp, e, c, b, or h instead of an upper-case format I, F, FP, E, C, B, or H.

Output example: Format E ␣1.23E–100␣

Space

Format e ␣1.23E–100

Space is deleted

- Only plus values are significant in format FP.

(3) Data and print output examples

Table 4-3 shows data and output examples.

Table 4-3 PRINT-Statement Output Example

Format	Data	Statement	Output
(None)	T=1234.45	PRINT␣T	123.45␣
	A$="ABCD"	DIM␣A$(5)	
		PRINT␣A$	ABCD␣␣
		PRINT␣A$(2)	C␣
	A$(Ø,)="AB"	DIM␣A$(3,2)	
	A$(1,)="CD"	PRINT␣A$(1,Ø)	C␣
	A$(2,)="EF"	PRINT␣A$(2,)	EF␣
I	T=1234.56	PRINT␣T:I6	␣␣1234␣
		PRINT␣T:I4	1234␣
		PRINT␣T:I3	***␣
F	T=–123.45	PRINT␣T:F6.1	–123.4␣
		PRINT␣T:F9.2	␣␣–123.45␣
		PRINT␣T:F9.3	␣–123.45Ø␣
	T=123456	PRINT␣T:F9.1	␣123456.Ø␣
		PRINT␣T:F5.1	*****␣
FP	T=123.45	PRINT␣T:FP6.1	␣123.4␣
		PRINT␣T:FP9.2	␣␣␣123.45␣
		PRINT␣T:FP9.3	␣␣123.45Ø␣
	T=123456	PRINT␣T:FP9.1	␣123456.Ø␣
		PRINT␣T:FP5.1	*****␣
E	T=–123.45	PRINT␣T:E1Ø.2	–1.23E2␣␣␣␣
		PRINT␣T:E13.5	–1.2345␣E2␣␣␣␣
		PRINT␣T:E15.7	–1.2345␣␣␣E2␣␣␣␣
	T=–Ø.12E1	PRINT␣T:E9.2	–1.2␣EØ␣␣␣␣
C	A$="F"	PRINT␣A$:C3	F␣␣␣
	A$="ABCDE"	DIM␣A$(5)	
		PRINT␣A$:C7	ABCDE␣␣␣
		PRINT␣A$:C3	ABC␣
		PRINT␣A$:C5	ABCDE␣
		PRINT␣A$(3):C3	D␣␣␣
	A$="ABCDEF"	DIM␣A$(6)	
		PRINT␣A$	ABCDEF␣
		PRINT␣A$(3)	D␣

Table 4-3 PRINT-Statement Output Example (Continued)

Format	Data	Statement	Output
B	I#=#1	PRINT␣I#:B1	1␣
		PRINT␣I#:B3	ØØ1␣
	I#=#1Ø11	DIM␣I#(4)	
		PRINT␣I#:B5	1Ø11␣␣
		PRINT␣I#:B3	Ø11␣
		PRINT␣I#(3):B3	1␣␣␣
		PRINT␣I#(Ø):B1	1␣
	I#=#1	PRINT␣I#	␣␣␣␣␣␣␣1␣
	I#=#1Ø11	DIM␣I#(4)	
		PRINT␣I#	1Ø11␣
	I#=#ØØØ1ØØ11	DIM␣I#(8)	
		PRINT␣I#	1ØØ11Ø1Ø␣
		PRINT␣I#(3)	1␣
	I#=#ØØØ1ØØ11	PRINT␣I#	␣␣␣1ØØ11␣
H	I#=#1	PRINT␣I#:H1	1␣
		PRINT␣I#:H2	␣1␣
	I#=#1Ø1Ø	DIM␣I#(4)	
		PRINT␣I#:H1	A␣
		PRINT␣I#:H2	A␣␣
	I#=#ØØØØ1Ø1Ø	DIM␣I#(8)	
		PRINT␣I#:H1	A␣
		PRINT␣I#:H2	␣A␣
	I#=#111Ø1Ø1Ø	DIM␣I#(8)	
		PRINT␣I#:H1	A␣
		PRINT␣I#:H2	EA␣
		PRINT␣I#(3):H1	1␣
		PRINT␣I#(3):H2	1␣␣
		PRINT␣I#(4):H1	Ø␣
		PRINT␣I#(4):H2	Ø␣␣
	I#=#ØØ11ØØ	DIM␣I#(6)	
		PRINT␣I#:H2	␣C␣
	I#=#11ØØ1Ø	PRINT␣I#:H2	32␣

Note

Example with the DIM statement means the array declaration is performed for the variable. If no DIM statement is marked, it means there is no array declaration for the variable.

Reverse display (PRINTR statement)

(1) Function

Edits data and displays the data on the screen in reverse mode.
See Section 4, "PRINT statement" for details.

(2) Format

```
PRINTR {variable [ : format]           } [,{variable [ : format]           }•••][;]
       {character-string-constant}    {character-string-constant}
```

The constant is displayed as is.   No line feed

Notes:

- Only characters of character codes 0 to 127 can be displayed in reverse mode.
  PRINTR containing other character displays has the same function as that of PRINT. In this case, PRINTR displays characters in normal mode.
- A line in which characters of character codes 128 to 255 are displayed cannot be displayed in reverse mode.
  In this case, PRINTR has the same function as that of PRINT, and it displays characters in normal mode.

Positioning the cursor (LOCATE statement)

(1) Function

This statements specifies the cursor position on the screen. (Referred to at the upper left on the screen)

(2) Format

```
LOCATE(m,n)
     m   →   column position (1 to 40)
     n   →   line position (1 to 20)
```

Note: Both m and n are numeric constants or variables.

Data statement (DATA statement)

(1) Function

This statement defines numeric, bit and character constant to be read with the RDATA statement.

(2) Format

```
DATA, constant, constant, ••••••••
```

Note: Any number of parameters maybe input in a DATA statement provided that it does not exceed two lines. Further, different types of constants may be input in a single DATA statement.

Reading data (RDATA statement)

(1) Function

This statement reads values from the DATA statement and assigns them to variables.

(2) Format

```
RDATA variable, variable, ••••••••
```

Notes:

- Any number of parameters maybe assigned in an RDATA statement provided that it does not exceed 2 lines. Further, different types of constants may be input in a single RDATA statement.
- If the definition type in the DATA statement and the type of the substituted variable are incompatible at data reading with the RDATA statement, ERROR W208 will be generated.

Read specification of data statement (RESTORE statement)

(1) Function

This statement specifies the data statement to be read with the RDATA statement.

(2) Format

```
RESTORE [line number or ∗ label]
```

Example :

```
 1ØØ  RESTORE 1ØØØ
 11Ø  FOR I=Ø TO 1Ø
 12Ø  RDATA A(I)
 13Ø  NEXT I
      :
1ØØØ  DATA Ø,1,3,7,9,11,13,17,19,23,29
```

Note: When the RESTORE-statement parameter is omitted, the first data statement is used.

Setting measurement parameters (PUT and WRITE 1000 statements)

(1) Function

Sets the spectrum analyzer measurement parameters from the PTA.
The same messages as those set by remote control are used.
This command is also used when sending inquiry messages to the spectrum analyzer.

(2) Format

```
PUT character constant or character variable
WRITE 1ØØØ, variable or character constant [,variable or character constant]
```

1) PUT statement
- A message of the same format as remote control is described in operands.
- Only a character constant or character variable can be described in the operands.
- Only one constant or variable can be described.
- The format cannot be specified.
- When a fixed value is set at all times, the program can be simplified using this statement.

Examples :

```
PUT " CF 5ØØMHZ"
```

→ Set measurement parameter center frequency to 500 MHz.

```
PUT " CF?"
```

→ Send measurement parameter center frequency inquiry message.

2) WRITE 1000 statement
- A message of the same format as remote control is described in operands.
- Variables or character constants can be described in the operands.
- Up to five constants or variables can be described.
- When variables are used, the format can be specified.
- This statement is effective when setting is performed several times with only part of the control message being changed and when values treated as variables are set values in the program.

Examples :

```
F=5ØØ
WRITE 1ØØØ,"CF ",F,"MHZ"
```

→ Set measurement parameter center frequency to 500 MHz.

```
WRITE 1ØØØ,"CF?"
```

→ Send measurement parameter center frequency inquiry message.

Measurement parameter/data read (GET, COM and READ 1000 statements)

(1) Function

Reads the spectrum analyzer measurement parameters and the measured result from the PTA.
The same messages as those set by remote control are used.

(2) Format

```
GET "inquiry command?",input variable
COM "inquiry command?">input variable[, input variable]
READ 1ØØØ, input variable[, input variable] or
READ 1ØØØ, input variable[;]
```

1) GET statement

- An inquiry command can be sent and the response data can be read with one statement. Only one inquiry command can be described in one statement.
- Only a character constants or character variables can be described in the "inquiry command" parameters. Only one constant or variable can be specified. The format cannot be specified.
- The response data is stored in the input variable. When the response data contains a character, a character variable is specified. When the response data is numeric (numeric character) only, it may be a numeric variable or a character variable.
- When the response data consists of multiple data separated by a ",", everything up to the last data is stored in one variable as one data. Therefore, when a character variable is specified, if the array size is too small, all the response data may not be stored.
- Only one input variable can be specified. A ";" cannot be specified at the end of the statement.
- When the same inquiry command is always sent, the program can be simplified using this statement.

Example :

```
GET "CF?", A$
```

→ Send the center frequency inquiry message and store the response data in input variable A$.

2) COM statement

- An inquiry command can be sent and the response data can be read with one statement. However, only one inquiry command can be described in one statement.
- Character constant or character variable or character constant and character variable can be specified in the "inquiry command" parameter.
  The format can also be specified for variables.

- The response data is stored in the input variable. When the response data contains a character, a character variable is specified. When the response data is numeric (numeric character) only, it can be a numeric variable or character variable.
- Multiple variables can be described. When the response data consists of multiple data delimited by a ",", the delimited data are stored sequentially in the specified variables.
  However, array variables cannot be used as input variables.
- A ";" cannot be specified at the end of the statement.
- This statement is effective when reading is performed several times with only part of the inquiry message changed and when sending an inquiry message for a value treated as a variable in the program.

Example :

```
I=1
COM "MKML? ", I>ML
```

→ Send the 1st marker level inquiry message of the multimarker, and store the response data to input variable ML.

Note: The inquiry message for each level of the multimarker is specified by "MKML? n " (n: multimarker No.). This statement is useful for reading the level of each marker by changing only the value of n.

3) READ 1000 statement

- This statement reads the response data only. Therefore, it is effective only when a PUT or WRITE 1000 statement is used to send an inquiry message.
- The response data is stored in the input variable. When the response data contains a character, a character variable is specified. When the response data is numeric (numeric character) only, it can be a numeric variable or character variable.
- Multiple input variables can be described. When the response data consists of multiple data delimited by a ",", the delimited data is stored sequentially in the specified variables.
- When the response data is treated as one data, even when it consists of multiple data delimited by a ",", the entire response, including the ",", can be stored in one variable by specifying ";" at the end of the statement. In this case, only one input variable can be specified. Data delimited by a "," can also be read by specifying only one variable without a ";" at the end and executing this statement repeatedly.
- When there is no response data, "***" is output.

Example :

```
WRITE 1ØØØ,"CF? "
READ 1ØØØ, A$
```

→ Store the response data to the center frequency inquiry command in A$.

Program loading and execution (CHAIN statement)

(1) Function

This statement loads and executes a file in memory card.

(2) Format

```
CHAIN "file name"
```

Note: The RUN, CONT or STEP commands (set in the execution state) remain valid even after the CHAIN command is executed. Consequently, the lines at which execution is suspended also remain effective.

ENABLE EVENT statement

(1) Function

Enables the specified interrupt.

When the specified interrupt occurs, the program will branch to the event interrupt subroutine defined by the ON EVENT statement.

(2) Format

```
ENABLE EVENT I/O number,event 3,event 2,event 1,event 0
```

Notes:

- There are 2 types of I/O numbers: numeric variables and numeric constants.
- Events 0 to 3 can be numeric variables and constants, bit variables and constants, or hexadecimal constants.
- This statement can be executed directly.
- Events 0 to 3 indicate 32 bits of I/O interrupt events as shown below.
- The defined bits (b0 to b31) are enabled when "1" and disabled when "0".
- When the master bit (b31) was set to "1", all the defined conditions are valid regardless of the value of bits b0 to b30.

b31 b24	b23 b16	b15 b8	b7 b0
Event 3	Event 2	Event 1	Event 0

(3) Types of I/O interrupts

(a) Time-specification interrupts

Three kinds of time-specification interrupts are available.

1) DELAY

Generates an event interrupt after the specified time has elapsed.
The time can be specified as a remote control command or by a PUT or WRITE statement.

DELAY setting

"EDLY t" t: 0 to 3600 (s) 1 sec resolution

- Time counting starts from the time set by this command.
- When the time is reset during counting, counting restarts.
- If t=0 was set, counting is interrupted.
- There is no set value t inquiry command.

2) Time

Generates an event interrupt at the specified time.
The time can be specified as a remote control command or by a PUT or WRITE statement.

Time setting
"ETIM t_1, t_2, t_3"

t_1: Specifies the hour. (0 to 23)
t_2: Specifies the minute. (0 to 59)
t_3: Specifies the second. (0 to 59)

- When the time is reset during counting, counting restarts.
- There are no set value t_1, t_2, and t_3 inquiry commands.

3) Cycle

Generates an event interrupt at the specified cycle (time).
The cycle can be specified as a remote control command or by a PUT or WRITE statement.

Cycle setting

"ECYC t" t: 0 to 3600 (s) 0.1 sec resolution

- If t=0 was set, time counting is interrupted.
- There is no set value t inquiry command.

(b) Soft keys and data knob interrupt

1) Soft keys ([F1] to [F5])

When a PTA menu (3/4) [F1] to [F5] key (corresponding to system variables EX1 to EX5) is pressed, an event interrupt is generated. This also applies to the PTA keyboard [F1] to [F5] keys.

2) Cursor control keys

When the PTA menu (2/4) [CURSOR UP : F2] key or [CURSOR DOWN : F3] key is pressed, an event interrupt is generated.

3) Data knob

When the data knob is turned, an event interrupt is generated.
However, when the spectrum analyzer measurement parameter setting is effective, an event interrupt is not generated.

Clockwise and counterclockwise revolution can be detected.

I/O type	I/O number	Contents
Clock (DELAY)	1	b31 | b0 b31: Master bit b0: Interrupt occurrence
Clock (TIME)	2	b31 | b0 b31: Master bit b0: Interrupt occurrence
Clock (CYCLE)	3	b31 | b0 b31: Master bit b0: Interrupt occurrence
SOFT KEY, data knob	11	b31 | b17 | b16 | b9 | b8 | b4 | b3 | b2 | b1 | b0 b0: [F1]Key b1: [F2]Key b2: [F3]Key b3: [F4]Key b4: [F5]Key b8: [CURSOR UP:F2]Key b9: [CURSOR DOWN:F3]Key b16: Data knob right b17: Data knob left b31: Master bit

DISABLE EVENT statement

(1) Function

Disables the specified interrupt.

(2) Format

```
ENABLE EVENT I/O number[,event 3,event 2,event 1,event 0]
```

Notes:

- There are 2 types of I/O number: numeric variables and numeric constants.
- Events 0 to 3 can be numeric variables and constants, bit variables and constants, or hexadecimal constants.
- Events 0 to 3 may be omitted. When omitted, all interrupt events will be disabled.
- This statement can be directly executed.
- The defined bits are disabled when "1" and retain their previous enable/disable state when "0". However, master bit (b31) setting is meaningless. (Don't care)

ON EVENT statement

(1) Function

Registers the subroutine to branch to when the specified interrupt event occurs.

(2) Format

```
ON EVENT I/O number, line number(or * label)
```

Notes:

- There are 2 types of I/O number: numeric variables and numeric constants.
- This statement can be executed directly.
- The function STATUS (M) is used as the interrupt event identifier. For more details, see Section 4, "Functions", (5) Dedicated functions.

RETINT statement

(1) Function

Returns from the event interrupt subroutine.

(2) Format

```
RETINT
```

Notes:

- If any other return command is executed to return from an event interrupt subroutine, an execution termination error (F243) will be generated.
- If the RETINT command is executed for other than event interrupt, an execution termination error (F251) will be generated.
- It is possible to branch to a normal subroutine (GOSUB … RETURN) from the event interrupt subroutine.

Character size specification (DCHSIZE statement)

(1) Function

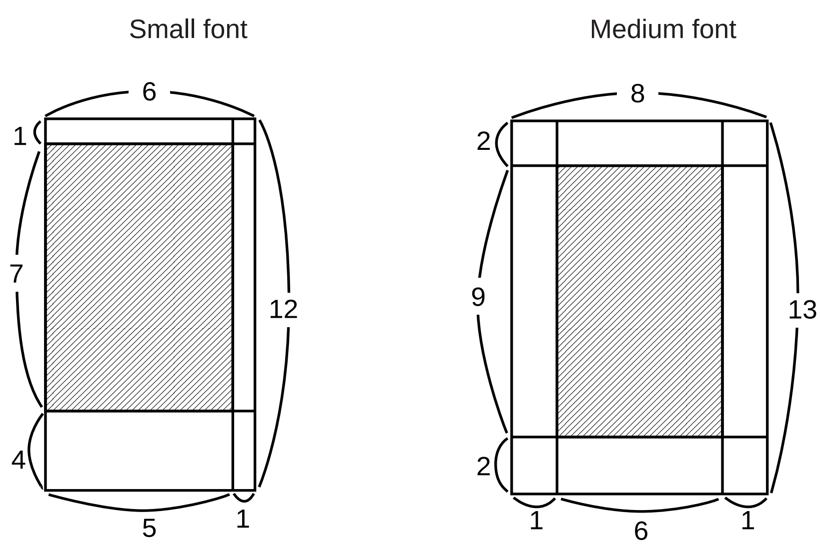
Specifies the display character size at system subroutine DCH execution.

(2) Format

```
DCHSIZE Character size number
```

Character size number	
0	Small font
1	Medium font

- The patterns of small/medium character fonts are shown below:



The units are dots on the CRT.

- The display character size can not be changed by PRINT statement, etc.
- Initialized by the RESET command.

Home position (HOME statement)

(1) Function

This statement moves the cursor to the home position (upper left).

(2) Format

```
HOME
```

Delete (ERASE statement)

(1) Function

This statement deletes statements after the line with the cursor.

(2) Format

```
ERASE
```

Note: When only the PTA screen is erased from the display, the screen is only partially erased. To erase the screen entirely, use the system subroutine CFL (see Section 5, "CFL subroutine").

Time wait (WAIT statement)

(1) Function

This statement is used to wait for a specified time period.

(2) Format

```
WAIT { Numeric variable }
     { Numeric constant }
```

Waiting time (unit: second, Ø.Ø1 s resolution)

System subroutine execution (CALL statement)

(1) Function

This statement is used to execute system subroutines.
For details of system subroutines, see Section 5, "System Subroutines".

(2) Format

```
CALL system subroutine name[(parameter[,parameter...])]
```

ON ERROR statement

(1) Function

Registers the subroutine to branch (interrupt) to when an error occurs.

(2) Format

```
ON ERROR line number(or *label)
```

Notes:

- Execution is halted when an error occurs during the execution of an error processing subroutine.
- If there is an error statement right after the line where the error occurred, only the error statement will be executed.
- If the error is an execution termination error, no interrupt will occur.
- If an error occurs during data input with the INPUT statement, no interrupt will occur.
- The function ERRREAD (m) identifies the error code and line the error occurred. For details, see Section 4, "Dedicated functions".
- Multiple interrupts with event interrupts are possible.
- The error occurred during an error interrupt processing is not applied.

OFF ERROR statement

(1) Function

Removes the registered subroutine to branch (interrupt) when an error occurs. No error interrupt will occur while after executing this command.

(2) Format

```
OFF ERROR
```

RETERR statement

(1) Function

Returns from an error interrupt.

Continues from the statement following the statement where the error occurred.

(2) Format

```
RETERR
```

Notes:

- If the RETURN or RETMAIN commands are used to return from an error interrupt ,an execution termination error (F243) will result.
- If the RETINT command is executed to return from an error interrupt, an execution termination error (F251) will result.
- If the RETERR command is executed when there is no error interrupt, an execution termination error (F252) will result.
- It is possible to branch to a normal subroutine (GOSUB … RETURN) from the event interrupt subroutine.

RETRY statement

(1) Function

Returns from an error interrupt.
Execution is retried from the statement on which error occurred.

(2) Format

```
RETRY
```

Notes:

- If the RETURN or RETMAIN commands are used to return from an error interrupt, an execution termination error (F243) will result.
- If the RETINT command is executed to return from an error interrupt, an execution termination error (F251) will result.
- If the RETRY command is executed when there is no error interrupt, an execution termination error (F252) will result.
- It is possible to branch to a normal subroutine (GOSUB … RETURN) from the event interrupt subroutine.

RESUME statement

(1) Function

Returns from an error interrupt.
Continues from the specified line.

(2) Format

```
RESUME line number(or *label)
```

Notes:

- If the RETURN or RETMAIN commands are used to return from an error interrupt, an execution termination error (F243) will result.
- If the RETINT command is executed to return from an error interrupt, an execution termination error (F251) will result.
- If a command other than the RESUME command is executed when there is no error interrupt, an execution termination error (F252) will result.
- It is possible to branch to a normal subroutine (GOSUB … RETURN) from the event interrupt subroutine.

GIVEUP statement

(1) Function

Returns from an error interrupt.
Halts program execution.

(2) Format

```
GIVEUP
```

Notes:

- If the RETURN or RETMAIN commands are used to return from an error interrupt, an execution termination error (F243) will result.
- If the RETINT command is executed to return from an error interrupt, an execution termination error (F251) will result.
- If the GIVEUP is executed when there is no error interrupt, an execution termination error (F252) will result.
- It is possible to branch to a normal subroutine (GOSUB … RETURN) from the event interrupt subroutine.

Error branch (ERROR statement)

(1) Function

To continue execution after warning-error generation, an ERROR statement can be used. Multiple lines can be used for ERROR statements.
See Section 8, "ERROR Statement" for details.

(2) Format

```
ERROR(error number,program line or *label to be executed next)
```

Error main (ERRMAIN statement)

(1) Function

This statement branches to the highest level routine when an error that allows execution to continue (error code beginning with the letter W) is generated while the program was running.

(2) Format

```
ERRMAIN(error number)
```

Notes:

- When an ERRMAIN statement was executed in the highest level routine, the error code becomes F213.
- See Section 8, "ERRMAIN Statement" for details.

Data input 1 (READ statement)

(1) Function

This statement is used to receive data from a device connected to the RS-232C or GPIB through the specified port.

(2) Format

```
READ address,input variable[,input variable....]
READ address,variable[;]
```

- When ";" is not added at the end of the statement, commas (",") in the received data are assumed to be data delimiters and are stored in each variable.
- When ";" is added at the end of the statement, commas (",") are not assumed to be data delimiters and everything up to the data terminator is stored in one variable.

Data input 2 (BREAD statement)

(1) Function

This statement is used to receive one byte of binary data from a device connected to the RS-232C or GPIB through the specified port. When the specified port is a device port, this statement cannot be executed.

(2) Format

```
BREAD address,input variable[,input variable....]
```

Data input 3 (WREAD statement)

(1) Function

This statement is used to receive one word of binary data from a device connected to the RS-232C or GPIB through the specified port. The data is stored in the input variable as high byte to low byte in sending order. When the specified port is a device port, this statement cannot be executed.

(2) Format

```
WREAD address,input variable[,input variable....]
```

Data output 1 (WRITE statement)

(1) Function

This statement sends data to a device connected to the RS-232C/GPIB/parallel (centronics) through the specified port.

(2) Format

```
WRITE address,variable[:format][,variable[:format]...][;]
```

- The output data can also be a character constant.
- When ";" is added at the end of the statement, a terminator is not output.
- The output destination depends on the addressing method and GPIB port mode (system controller/device).

Data output 2 (BWRITE statement)

(1) Function

This statement sends one byte of binary data to a device connected to the RS-232C/GPIB/parallel (centronics) through the specified port. When the specified port is a device port, this statement cannot be executed.

(2) Format

```
BWRITE address,variable[,variable...]
```

Notes:

- Neither format nor ";" can be specified.
- The terminator is not output.

Data output 3 (WWRITE statement)

(1) Function

This statement sends one word (two bytes) of binary data in order of high byte to low byte to a device connected to the RS-232C/GPIB/parallel (centronics) through the specified port. When the specified port is a device port, this statement is not executed.

(2) Format

```
WWRITE address,variable[,variable...]
```

Notes:

- Neither format nor ";" can be specified.
- The terminator is not output.
- When a one- or two-digit value is used (e.g. 5 or 17) for an address, the value becomes the address of the device connected to the port specified by the PORT command as a remore control command (Indirect Port specification). However, when a three-digit value (e.g. 105 or 217) is used, the first digit becomes the port address and the lower two digits become the address of the device connected to the port (Direct Port specification).
- The lower two digits of the address at indirect or direct port specification have no meaning in the RS-232C and parallel (centronics). However, these digits should still be specified for form' s sake.

  Example:

`WRITE␣5,"ABC"`	Data is sent to address 5 through the port specified by the PORT command (indirect port specification).
`READ␣1ØØ,A$`	Data is input from a device connected to port No. 1 (RS-232C) (direct port specification).
`WRITE␣2Ø5,"ABC"`	Data is sent to address 5 through port No. 2 (GPIB) (direct port specification).
`WRITE␣3ØØ,"ABC"`	Data is sent to a device connected to port No.3 (parallel (centronics)) (direct port specification).

These address specifications are effective for the WRITE, BWRITE, WWRITE, READ, BREAD, WREAD and LISTG statements.
The relationship between the port specification command and controller port is as follows:

	Indirect port specification	Direct port specification		
	WRITE 5	WRITE 1Ø5	WRITE 2Ø5	WRITE 3Ø5
At power-ON or after "PORT␣1" execution	*1 The RS-232C port is a controller port.	*1 The RS-232C port is a controller port.	The GPIB port is a controller port.	*1 The paralell (centronics) port is the controller port.
After "PORT␣2" execution	The GPIB port is a controller port.	*1 The RS-232C port is a controller port.	The GPIB port is a controller port.	*1 The parallel (centronics) port is the controller port.
After "PORT␣3" execution	*1 The parallel (centronics) port is the controller port.	*1 The RS-232C port is the controller port.	The GPIB port is the controller port.	*1 The parallel (centronics) port is the controller port.

*1: Addresses specified in the RS-232C, parallel (centronics) have no meaning. However, these addresses should still be specified for form' s sake.

Data writing to the dual port memory (WDPM statement)

(1) Function

This statement writes data to the dual port memory.
See Section 7, "Dual Port Memory" for details.

(2) Format

```
WDPM memory number,variable[:format][,variable[:format]....]
```

Notes:

- The output data can also be character constants.
- ";" cannot be specified.
- This statement can be executed regardless of the GPIB mode (system controller/device).

Data reading from the dual port memory (RDPM statement)

(1) Function

This statement reads data from the dual port memory.
See Section 7, "Dual Port Memory" for details.

(2) Format

```
RDPM memory number,input variable[,input variable ....]
```

- ";" cannot be specified.
- When data delimited by "," is input, multiple input variables are specified.

S.O.S (SOS)

(1) Function

This statement is displayed in the statement where a syntax error is generated during program loading.

(2) Format

```
SOS
```

Notes:

- A statement with SOS added is treated as a comment statement, the same as a REM statement, but when the program is run, it is treated as a syntax error.
- Line-number errors are treated as syntax errors (W6) and SOS is not displayed.

Setting the pseudorandom number sequence (RNDMIZE statement)

(1) Function

Sets a new initial value of a pseudorandom number sequence generated by the RND function.

(2) Format

```
RNDMIZE
```

Note: If this statement is not executed, the RND function in the program generates the same pseudo-random number sequence each time the program is executed.

Calling the PTA library (CALLIB statement)

(1) Function

This statement calls the specified PTA library.

(2) Format

```
CALLIB "PTA library name" [,parameter]
```

parameter: Numeric variable or constant (up to 10 parameters)

PTA library name: Alphanumeric string with up to 8 characters starting with a capital alphabet
Characters available for the 2nd character on :
- Under bar
- Capital alphabet : A to Z
- Small alphabet : a to z
- Numeral : 0 to 9

However, small alphabets are converted to capitals.

- The specified PTA library is called out. When the STOP statement is executed in the called PTA library, the system returns to the program where the CALLIB statement was executed.
- Up to 10 parameters can be sent to the called PTA library. In this case, parameter values are assigned to the local variables specified by the PARASET statement of the called PTA library. (See PARASET.)
- Nesting of the PTA library by the CALLIB statement is available up to 10 times.

Note: The PTA library, from the start line to the STOP statement, is counted as one program unit. (The STOP statement may come in the middle of the program.) The CALLIB statement calls this program unit.

Removing the PTA library from program memory (REMOVE statement)

(1) Function

This statement removes the specified PTA library from the program memory.

(2) Format

```
REMOVE ["PTA library name"]
```

Alphanumeric string with up to 8 characters starting with a capital alphabet
Characters available for the 2nd character on :
Under bar
Capital alphabet : A to Z
Small alphabet : a to z
Numeral : 0 to 9
However, small alphabets are converted to capitals.

- The specified PTA library is removed from the program memory. However, it is not possible to specify the PTA library in execution (or an error is generated if specified).
- When this function is directly executed without specifying a program, all the PTA libraries in the memory are removed.
- When the PTA library specified as the object of the program execution and edition commands is removed by the EDITLIB command, the specification of the EDITLIB command is cleared.

Clearing common variables (COMCLEAR statement)

(1) Function

This statement clears all the common variables residing in the memory.

(2) Format

```
COMCLEAR
```

- All the common variables residing in the memory are cleared.
- When this statement is executed in the nested PTA library, an error is generated.

Setting CALLIB parameter values (PARASET statement)

(1) Function

This statement sets the parameter values sent from the CALLIB statement to the specified local variables.

(2) Format

```
PARASET Parameter[,parameter]
```

Up to 10 real-number local variables

- Parameters sent from the side that called the PTA library are set to local variables. Only the real-number local variable can be used. When common and other variables are specified, an error is generated at input. When the call side of the PTA library does not send parameters, the variable value is set to be zero.

Loading the PTA library file LOADLIB statement)

(1) Function

This statement loads the function-specified PTA library file.

(2) Format

```
LOADLIB "File name"
```

Alphanumeric string with up to 6 characters starting with a capital alphabet
Characters available for the 2nd character on :

Capital alphabet	: A to Z
Small alphabet	: a to z
Numeral	: 0 to 9

However, small alphabets are converted to capitals.

- The PTA library file saved in the memory card is loaded. If a PTA library named the same as one already existing in the memory is loaded, the content of the existing PTA library is replaced with that of the newly loaded PTA library.
- It is not possible to load the file in which a PTA library named the same as one in execution is saved.

SECTION 5

EXTENDED PTL

TABLE OF CONTENTS

SECTION 5 EXTENDED PTL

There are system variables, system functions, and system subroutines in the extended PTL.

The extended PTL can execute operations and evaluation of measurement results, and control external devices.

System Variables

PTA provides system variables with pre-defined names in addition to user-defined variables. Using these system variables, the measured data can be read.

Variable name	Number of array elements	Purpose	Data meaning	Read/ Write
EX1	- - -	Corresponding to F1 key	Numbers 0 and 1 are switched alternately each time the F1 key is pressed.	R/W
EX2	- - -	Corresponding to F2 key	Numbers 0 and 1 are switched alternately each time the F2 key is pressed.	R/W
EX3	- - -	Corresponding to F3 key	Numbers 0 and 1 are switched alternately each time the F3 key is pressed.	R/W
EX4	- - -	Corresponding to F4 key	Numbers 0 and 1 are switched alternately each time the F4 key is pressed.	R/W
EX5	- - -	Corresponding to F5 key	Numbers 0 and 1 are switched alternately each time the F5 key is pressed.	R/W
EX6	- - -	Corresponding to etc key of each hierarchy	0 to 3: Switches a PTA function key hierarchy (*)	R/W

* Soft-key menus can be changed by inputting 0, 1, 2 and 3 to the system variable EX6, as shown below.

However, EX6 is disabled when the PTA menus are not being executed.

Variable name	Number of array elements	Purpose	Data meaning	Read/ Write
DTØ	---	Time setting/reading (year: Gregorian calendar)	1960 to 2059	R/W
DT1	---	Time setting/reading (month)	0 to 12	R/W
DT2	---	Time setting/reading (date)	0 to 31	R/W
DT3	---	Time setting/reading (hour)	0 to 23	R/W
DT4	---	Time setting/reading (minute)	0 to 59	R/W
XMA	501	Waveform memory of TRACE-A	Waveform data in 0.01dBm unit	R/W
XMB	501	Waveform memory of TRACE-B	Waveform data in 0.01dBm unit	R/W
XMG	501	Waveform memory of TRACE-BG	Waveform data in 0.01dBm unit	R/W
XMT	501	Waveform memory of TRACE-Time	Waveform data in 0.01dBm unit	R/W
SMA	501	Submemory A	–32768 to 32767: 2-byte integer/1 point	R/W
SMB	501	Submemory B	–32768 to 32767: 2-byte integer/1 point	R/W
SMT	501	Submemory Time	–32768 to 32767: 2-byte integer/1 point	R/W
IMA	501	Image memory A	–32768 to 32767: 2-byte integer/1 point	R/W
IMB	501	Image memory B	–32768 to 32767: 2-byte integer/1 point	R/W
RMA	501	Real number memory A	8-byte floating point real number/1 point	R/W
RMB	501	Real number memory B	8-byte floating point real number/1 point	R/W

	EX6 = 0	EX6 = 1	EX6 = 2	EX6 = 3
F1	RUN	PLIST	F1 *	YES
F2	STOP	CURSOR UP	F2 *	NO
F3	CONT	CURSOR DOWN	F3 *	(None)
F4	RESET	LOAD	F4 *	(None)
F5	PTA OFF	RUN	F5 *	(None)
F6	etc (1/4)	etc (2/4)	etc (3/4)	etc (4/4)

* Display cheracters can be defined with DEF subroutine.

System Subroutines

The MS2665C/67C/68C PTA has dedicated subroutines, called the system subroutines, executed by the CALL statement.

The system subroutines are shown below :

■ Display subroutines

• Displayed item erase :	`CALL CER(M)`
• Screen restore :	`CALL CRN(M)`
• Screen erase :	`CALL CFL(M)`
• Character-string display :	`CALL DCH(X,Y,text,M[,N])`
• Straight-line display :	`CALL DLN(XØ,YØ,X1,Y1,M[,N])`
• Square display :	`CALL DRC(XØ,YØ,X1,Y1,M[,N])`
• Circle display :	`CALL DCR(X,Y,R,M[,N])`
• Arc-line display :	`CALL DAR(XØ,YØ,RØ,W1,W2,M1[,M3])`
• Soft-key label registration:	`CALL DEF(M,text)`

■ File-operation subroutines

• File open (read) :	`CALL OPNI␣character string variable (or character constant)`
• File open (write) :	`CALL OPNO␣character string variable (or character constant)`
• File delete :	`CALL FDEL␣character string variable (or character constant)`
• Data load :	`CALL DALD variable`
• Data save :	`CALL DASV variable`
• File close :	`CALL CLS`

■ GPIB subroutine (GPIB port only)

- Interface clear : `CALL IFC`
  (Changeover to system controller port)
- Service request : `CALL RSV(M)`
- Take controller : `CALL TCT(M)`
- Changeover to device port : `CALL DEV`

■ Interface subroutine

- Status byte reading : `CALL GST(port number,address,input variable)`
- Interface control : `CALL GPIB(port number,control item number)`

■ Panel subroutines

- Front-panel operation lock : `CALL PNLL(Ø)`
- Front-panel operation lock cancellation : `CALL PNLU(Ø)`

■ Waveform memory subroutine

- Memory copy : `CALL COPY(MØ,M1)`
- Data conversion : `CALL CONV(K,MØ,M1,PØ,P1[,D])`
- Frequency axis logarithm conversion : `CALL SWLG(K,MØ,M1)`

NOTES

If parameters specified in each subroutine are outside the specified range, an error occurs and no graphic data is plotted.

CER and CRN subroutines

(1) Function

The CER/CRN subroutines perform erasure and display restoration of the character string, graph, scale, marker, etc. on the CRT screen.

(2) Format

```
CALL␣CER(MØ)  ······  Erases items MØ
CALL␣CRN(MØ)  ······  Restores items MØ display
```

M0	Item
0	Marker frequency, level, AT, RB
1	RLV, ST, VB
2	Frequency
3	Menu, data input area
4	Sweep marker
5	Scale line, Y-axis scale
6	Waveform
7	Markers, zone
8	Message in scale
9	Title, trace item, trigger switch, sweep status
10	All items above

Notes:

- See Section 1, "Screen Configuration of PTA" for the screen details.
- A numeric constant or numeric variable is used for M0.
- When clear/display return was performed with this subroutine, the state is held until it is reset by this subroutine or until the PTA is turned off.

CFL subroutine

(1) Function

This subroutine erases display items of each frame constituting the screen.

(2) Format

```
CALL␣CFL(M1)
```

M1 (Frame No.)	Display item
0	Waveform background
1	PTA screen
2	Scale line
3	Waveform display 2
4	Waveform display 3
5	Parameter
6	Display line
7	Trigger indicator
8	Marker zone
9	Template/mask standard line
10	Multi-marker No.
11	(Not used)
12	Marker/marker value display
13	Menu background
14	Menu characters
15	Setting and parameter characters, error message

Notes:

- A numeric constant or numeric variable is used for M1.
- This subroutine temporarily clears the screen.
  Therefore, when the display condition is reestablished; for example, when measurement parameter values are changed, or when characters and patterns are displayed; they are displayed.
- See Section 1, "Screen Configuration of PTA" for the screen details.

DCH subroutine

(1) Function

Displays a character string. (Referred to at the bottom left on the screen)

(2) Format

```
CALL␣DCH(X,Y,text,M1[,M2])
```

- M2: Character font specifications (normal/reversed display)
- M1: Screen specification
- text: Character string to be displayed
- Y: Y coordinate of first character
- X: X coordinate of first character

M1 (Frame No.)	Display item
0	Waveform background
1	PTA screen
2	Scale line
3	Waveform display 2
4	Waveform display 3
5	Parameter
6	Display line
7	Trigger indicator
8	Marker zone
9	Template/mask standard line
10	Multi-marker No.
11	(Not used)
12	Marker/marker value display
13	Menu background
14	Menu characters
15	Setting and parameter characters, error message

M2	Display mode
0	Normal display
1	Reverse diaplay

■ Range of each parameter

Font	First X coordinate (X)	First Y coordinate (Y)	Maximum No. ogf characters of string (text)
Small font	0 to 314	0 to 228	54
Medium font	0 to 312	0 to 227	40

Notes:

- The first X coordinate and Y coordinate specify the lower-left corner of the character.
- Numeric constants or numeric variables are used for X, Y, M1, and M2. "text" is a character constant or character variable.
- M2 is omissible and it is assumed to be 0 if omitted.
- The character size (small font/medium font) can be set with the DCHSIZE statement.
  DCHSIZE 0: Small font
  DCHSIZE 1: Medium font
- See Section 1, "Screen Configuration of PTA" for the screen details.

DLN subroutine

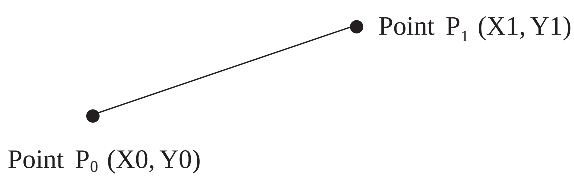
(1) Function

This subroutine displays a straight line (sectional line).

(2) Format

```
CALL␣DLN (XØ,YØ,X1,Y1,M1[,M3])
```

- M3: Line type
- M1: Screen type
- Y1: Y coordinate at point P1 (Ø to 239)
- X1: X coordinate at point P1 (Ø to 319)
- YØ: Y coordinate at point PØ (Ø to 239)
- XØ: X coordinate at point PØ (Ø to 319)



M1 (Frame No.)	Display item
0	Waveform background
1	PTA screen
2	Scale line
3	Waveform display 2
4	Waveform display 3
5	Parameter
6	Display line
7	Trigger indicator
8	Marker zone
9	Template/mask standard line
10	Multi-marker No.
11	(Not used)
12	Marker/marker value display
13	Menu background
14	Menu characters
15	Setting and parameter characters, error message

M3	Line type
0	Displays solid line
1	Erases solid line
2	Displays dashed line
3	Erases dashed line

Notes:

- A numeric constant or numeric variable is used for X0, Y0, X1, Y1, M1, and M3.
- M3 is omissible and it is assumed to be 0 if omitted.
- See Section 1, "Screen Configuration of PTA" for coordinate details.

DRC subroutine

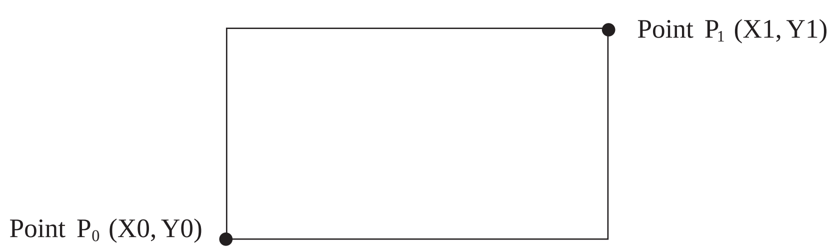
(1) Function

This subroutine displays a square based on a diagonal line between two specified points.

(2) Format

```
CALL␣DRC(XØ,YØ,X1,Y1,M1[,M3])
```

- M3: Line type
- M1: Screen type
- Y1: Y coordinate at point P1 (Ø to 239)
- X1: X coordinate at point P1 (Ø to 319)
- YØ: Y coordinate at point PØ (Ø to 239)
- XØ: X coordinate at point PØ (Ø to 319)



M1 (Frame No.)	Display item
0	Waveform background
1	PTA screen
2	Scale line
3	Waveform display 2
4	Waveform display 3
5	Parameter
6	Display line
7	Trigger indicator
8	Marker zone
9	Template/mask standard line
10	Multi-marker No.
11	(Not used)
12	Marker/marker value display
13	Menu background
14	Menu characters
15	Setting and parameter characters, error message

M3	Line type
0	Displays solid line
1	Erases solid line
2	Displays dashed line
3	Erases dashed line

Notes:

- A numeric constant or numeric variable is used for X0, Y0, X1, Y1, M1, and M3.
- M3 is omissible and it is assumed to be 0 if omitted.
- See Section 1, "Screen Configuration of PTA" for coordinate details.
- No display is performed if P0 (X0, Y0) and P1 (X1, Y1) are at the same axis.

DCR subroutine

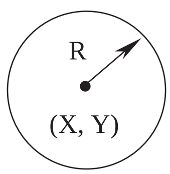
(1) Function

This subroutine displays a circle.

(2) Format

```
CALL␣DCR(X,Y,R,M1[,M3])
```

- M3: Line type
- M1: Screen type
- R: Radius (1 to 675)
- Y: Y coordinate of center (–239 to 478)
- X: X coordinate of center (–319 to 638)



M1 (Frame No.)	Display item
0	Waveform background
1	PTA screen
2	Scale line
3	Waveform display 2
4	Waveform display 3
5	Parameter
6	Display line
7	Trigger indicator
8	Marker zone
9	Template/mask standard line
10	Multi-marker No.
11	(Not used)
12	Marker/marker value display
13	Menu background
14	Menu characters
15	Setting and parameter characters, error message

M3	Line type
0	Displays solid line
1	Erases solid line
2	Diaplsys dashed line
3	Erases dashed line

Notes:

- Numeric constants or numeric variables are used for X, Y, R, M1, and M3.
- M3 is omissible and it is assumed to be 0 if omitted.
- See Section 1, "Screen Configuration of PTA" for coordinate details.

DAR subroutine

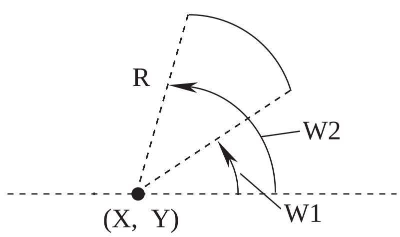
(1) Function

Displays an arc.

(2) Format

```
CALL␣DAR(X,Y,R,W1,W2,M1[,M3])
```

- M3 — Line type
- M1 — Screen type
- W2 — Plot end angle (–18Ø.ØØ to 18Ø.ØØ deg.)
- W1 — Plot start angle (–18Ø.ØØ to 18Ø.ØØ deg.)
- R — Radius (1 to 675)
- Y — Coordinate Y of center (–239 to 478)
- X — Coordinate X of center (–319 to 638)



M1 (Frame No.)	Display item
0	Waveform background
1	PTA screen
2	Scale line
3	Waveform display 2
4	Waveform display 3
5	Parameter
6	Display line
7	Trigger indicator
8	Marker zone
9	Template/mask standard line
10	Multi-marker No.
11	(Not used)
12	Marker/marker value display
13	Menu background
14	Menu characters
15	Setting and parameter characters, error message

M3	Line type
0	Displays solid line
1	Erases solid line
2	Displays dashed line
3	Erases dashed line

Notes:

- Numeric constants or numeric variables are used for the X, Y, R, W1, W2, M1, and M3.
- M3 is omissible and it is assumed to be 0 if omitted.
- See Section 1, "Screen Configuration of PTA" for coordinate details.

DEF subroutine

(1) Function

Registers a menu label (name) in the soft key menu.
When the PTA menu (3/4) is displayed, the labels registered by this subroutine are displayed.

(2) Format

```
CALL␣DEF(M,text)
        |  |
        |  └── Name of 3Ø characters maximum
        └───── Soft-key number (1 to 6)
```

Notes:

- M is a numeric constant or numeric variable.
- "text" is a character constant or character variable.
- The labels registered by this subroutine remain valid until the PTA is turned off.

OPNI, OPNO and FDEL subroutines

(1) Function

Opens a data file to write data to and read data from a memory card and deletes an existing data file.

(2) Format

```
CALL␣OPNI␣character string-variable(or character constant)
Open data read
CALL␣OPNO␣character string-variable(or character constant)
Open data write
CALL␣FDEL␣character string-variable(or character constant)
Delete data file
```

Notes:

- The data file name always begins with a % symbol and is followed by 6 or less alphanumeric characters including %.
- Do not remove the memory card while opening the data file in it.
- This subroutine cannot be used with the PTA program/library files on the memory card.

DALD and DASV subroutines

(1) Function

The DALD subroutine reads data saved in the memory card, and the DASV subroutine saves data to the memory card.

(2) Format

```
CALL␣DALD␣input variable:Read data from data file
CALL␣DASV␣variable      :Write data to data file
```

Notes:

- Data files are created as sequential files.
  Therefore, read them in the order in which they were written.
- Different types of data (for example, numeric type and character type) can be stored in one data file. However, when the type when the data was written and the type of input variable when the data was read cannot be assigned, an error is generated.

(3) Program example

```
 1Ø REM "*** DATA FILE ***"
 2Ø CALL OPNO"%DATA"
 3Ø FOR C=2 TO 1Ø
 4Ø D=C*C
 5Ø CALL DASV D
 6Ø NEXT C
```

Writing into file

```
 7Ø CALL CLS
 8Ø CALL OPNI"%DATA"
 9Ø FOR C=2 TO 1Ø
1ØØ CALL DALD D
11Ø PRINT "RES.=",D
12Ø NEXT C
13Ø CALL CLS
14Ø STOP
```

Reading from file

Excuted result

```
RES.=4
RES.=9
RES.=16
RES.=25
RES.=36
RES.=49
RES.=64
RES.=81
RES.=1ØØ
```

CLS subroutine

(1) Function

This subroutine closes the open data file.
Used for both write and read.

(2) Format

```
CALL␣CLS
```

IFC subroutine

(1) Function

When this subroutine is executed, the GPIB port becomes the system controller and outputs an "interface clear" signal to devices connected to the GPIB bus.

(2) Format

```
CALL␣IFC
```

Note: When CALL␣IFC is executed from the PTA, GPIB becomes the "connection port for peripheral devices" of the conditions for interface port connection. Accordingly, if GPIB has been set as the connection port for the external controller and the printer/plotter, the "connection port for the external controller" and the "connection port for the printer/plotter" becomes "no connection (NONE)".

OPNITF, OPNOTF, FDELTF subroutines

(1) Function

Opens a text file to write and to read text data from a memory card and deletes an existing text data file. This file can be read and written as plain text file on personal computer. File attribute (.txt) is added automatically.

(2) Format

`CALL␣OPNITF`	character string-variable (or character constant)
	Open text data read
`CALL␣OPNOTF`	character string-variable (or character constant)
	Open text data write
`CALL␣OPNOTF`	character string-variable (or character constant)
	Delete data file

Notes:

- The text data file name is followed by 6 or less alphanumeric characters.
- Do not remove the memory card while opening the next data file in it.
- This subroutine cannot be used with the PTA program/library files on the memory card.

DALDTF, DASATF subroutines

(1) Function

The DALD subroutine reads text data saved in the memory card, and the DASV subroutine saves data to the memory card.

(2) Format

CALL␣DALDTF	string-variable :	Read data from text data file
CALL␣DASVTF	string-variable (character constant) :	Write data to next data file

Notes:

- When DALDTF subroutine is executed, 1 line is read from text data file, and that is stored string-variable. If the text data is longer than variable length, the text data is cut by variable length.
  If string-variable is not used, an error is generated.
- When DASVTF subroutine is executed, 1 line is text data is stored to data file. If string-variable is not used, an error is generated.

(3) Program example

```
10   CALL OPNOTF "RWTEST"
20   FOR I=0 TO 25
30   D$=CHR$ (64+I)
40   CALL DASVTF D$
50   NEXT I
60   CALL CLSTF
70   FOR I=0 TO 25
80   CALL DALDTF D$
90   PRINT D$
100  NEXT I
110  CALL CLSTF
120  STOP
```

CLSTF subroutine

(1) Function

This subroutine closes the open data file.
Used for both write and read.

(2) Format

```
CALL␣CLSTF
```

RSV subroutine

(1) Function

This subroutine sends the service request to the controller when the GPIB port (the first interface) is used as a device port.

(2) Format

```
CALL␣RSV(M)
```

M	PTA Event Status Register (MSB)							(LSB)
0	×	×	×	×	0	0	0	1
1	×	×	×	×	0	0	1	0
2	×	×	×	×	0	0	1	1
3	×	×	×	×	0	1	0	0
4	×	×	×	×	0	1	0	1
5	×	×	×	×	0	1	1	0
6	×	×	×	×	0	1	1	1
7	×	×	×	×	1	0	0	0
8	×	×	×	×	1	0	0	1
9	×	×	×	×	1	0	1	0

(× means don't-care bit which does not change.)

The PTA event status register is defined as the extended status of Status-Byte bit 1.

Therefore, setting the left-described data (into the PTA Even Status Register) indirectly sets Status-Byte bit 1 as a summary bit.

The RQS bit (bit 6) is set as the logical AND of each Status-Byte bits to issue a service request to the controller.

The GPIB commands (used to read the Status Byte and PTA Event Status Register from the external controller) are *STB? and ESR1 ?, respectively.

Notes:

- A numeric constant or numeric variable is used for M.
- This subroutine is effective only when the GPIB port is connected with the external controller (the device port mode).

TCT subroutine

(1) Function

This subroutine causes controlling right to be passed to another device provided that the GPIB port is used as a system controller port.

(2) Format

```
CALL␣TCT(M)
```

Address of device to which control right is passed.

Notes:

- M is the GPIB address from 0 to 30, and a numeric constant or numeric variable is used.
- This subroutine is effective only when the GPIB port is a system controller port.

DEV subroutine

(1) Function

This subroutine causes the GPIB port to become a device port when it has previously been used as the system controller.

(2) Format

```
CALL␣DEV
```

Note: When the CALL DEV subroutine is executed from PTA, the "connection port for the external controller" of the conditions for interface port connection becomes GPIB. Accordingly, if GPIB has been set as the connection port for peripheral devices and the printer/plotter, the "connection port for peripheral devices" and the "connection port for the printer/plotter" becomes "no connection (NONE)".

GST subroutine (GST)

(1) Function

When the GPIB port is set as the connection port for the external controller, a serial poll is executed to the device specified by address, and the status value is read and stored as an input variable.

(2) Format

```
CALL␣GST(P,address,input variable)
```

- address: GPIB address on the device (Ø to 3Ø)
- P: Specified port (2: GPIB)

Notes:

- The read status value will be stored in the input variable. Input variable can be either a real-number, integer, or bit type variable.
- This subroutine is effective only when the GPIB port is a system controller port.
- This subroutine cannot be executed on the RS-232C/parallel (centronics).

Interface control subroutine (GPIB and RS-232C)

(1) Function

The "Interface Clear" (IFC), "Remote" (REN), "Local" (LCL), "Device Clear" (DCL), "Local Rockout" (LLO), and "Device Trigger" (DTR) are sent, and "Return to Local" (RTL) is set from the specified port.

(2) Format

`CALL␣GPIB(P,Ø)`	Sends IFC
`CALL␣GPIB(P,1[,address])`	Sends REN
`CALL␣GPIB(P,2)`	Sends RTL
`CALL␣GPIB(P,3[,address])`	Sends LCL
`CALL␣GPIB(P,4[,address])`	Sends DCL
`CALL␣GPIB(P,5)`	Sends LLO
`CALL␣GPIB(P,6,address)`	Sends DTR

P : Specified port No. (RS-232C: 1, GPIB: 2, Parallel (centronics): 3)

Address: GPIB device address of Ø to 3Ø

Notes:

- P and address are numeric constants or numeric variables.
- The actions of each subroutine are described below.

IFC :
- The IFC line is turned on for 100 É sec. The interface functions of all connected devices are initialized.
- Initialization is executed only for the corresponding interface functions. This code does not affect device functions.
- All talkers and listeners are not released.
- This does not affect the SRQ line.
- If the system passes control of the GPIB port to other controllers with the CALL TCT (m) command, control will be automatically returned to the PTA when execution is finished.
- This subroutine terminates normally without performing any processing for the RS-232C.

REN:
- When [, address] is omitted, the REN line is turned ON. Afterwards when the device is set to listener, it will assume remote control status.
- When [, address] is specified, the REN line is turned on. The device specified by [, address] will be identified as the listener and assume remote control status.
- Can be executed only when the specified port is a system controller port.
- This subroutine terminates normally without performing any processing for the RS-232C.

Notes: (Continued)

RTL:
- When the GPIB port is identified as the device, the PTA assumes the local control status. (This has the same effect as pressing the [LOCAL] key.)
- Only "2" can be specified as the port No.

LCL:
- When [, address] is omitted, the REN line is turned off. All devices assume local control status.
- When [, address] is specified, all listeners are released. After that, the device specified by [, address] is selected as the listener and assumes local control status. The REN line does not change.
- Can be executed only when the specified port is a system controller port.

DCL:
- When [, address] is omitted, "DCL" is sent and all device functions on the GPIB are initialized.
- When [, address] is specified, (Selected Device Clear) is sent and the device function specified by [, address] is initialized.
- Can be executed only when the specified port is a system controller port.

LLO:
- Disables the remote to local switching function of all devices on the GPIB. You will not be able to switch the device to local with the [Local] key on the panel.
- Switching is possible with the REN and LCL commands from the PTA.
- This mode can be exited with the LCL command in which the [, address] is omitted.
- Can be executed only when the specified port is a system controller port.

DTR:
- Triggers the specified device. The specified device begins the predetermined operation.
- Can be executed only when the specified port is a system controller port.
- This subroutine terminates normally without performing any processing for the RS-232C.

PNLU and PNLL subroutine

(1) Function

Sets LOCK/UNLOCK of the front panel when PTA is on.

(2) Format

```
CALL␣PNLU(Ø)      unlocks front panel.
CALL␣PNLL(Ø)      Locks front panel.
```

Note: The front-panel soft keys [F1] to [F6], [Shift], [Local] ,and numeric keys cannot be lock-out.

COPY subroutine

(1) Function

This subroutine copies the data in a specified waveform memory (copy source) to another waveform memory (copy destination). For example, use of the sub memory permits measurement in parallel with data processing.

(2) Format

```
CALL␣COPY(MØ,M1)
          │  └─Destination memory
          └────Source memory
```

M0, M1	Memory	System variable name	Type
0	Measurement memory	XMA ( )	Integer (0.01 dBm unit)
1	Measurement memory	XMB ( )	Integer (0.01 dBm unit)
2	Submemory a	SMA ( )	Integer (0.01 dBm unit)
3	Submemory b	SMB ( )	Integer (0.01 dBm unit)
4	Image memory a	IMA ( )	Integer
5	Image memory b	IMB ( )	Integer
6	Real number memory a	RMA ( )	Real number
7	Real number memory b	RMB ( )	Real number
8	Measurement memory	XMT ( )	Integer
9	Measurement memory	XMB ( )	Integer
10	Sub memory	SMT ( )	Integer

Notes:

- M0 contents are copied in M1. M0 contents are not changed. Previous contents of M1 are lost.
- A numeric constant or numeric variable is used for M0 and M1.
- Data cannot be copied between integer memory and real number memory.

CONV subroutine

(1) Function

This subroutine converts the measurement data of the measurement memory and performs the operation between memories.

(2) Format

```
CALL␣CONV(K,MØ,M1,PØ,P1[,D])
            |  |  |  |     |
            |  |  |  |     └── Sub-data
            |  |  |  └──────── Area specification (PØ<P1)
            |  |  └─────────── Destination memory
            |  └────────────── Source memory
            └───────────────── Function specification
```

K				Conversion (operation) function
0	Integer (0.01 dBm)		→	Real number (dBm)
1	Real number (dBm)		→	Integer (0.01 dBm)
2	Integer (0.01 dBm)		→	Real number (mW) M1 (x)=10 ↑ (M0 (x)/1000)
3	Real number (mW)		→	Integer (0.01 dBm) M1 (x)=INT (1000*LOG_{10} (M0 (x)))
4	ADD	M1=M0+D		
5	SUB	M1=M0–D		
6	MUL	M1=M0*D		
7	DIV	M1=M0/D		
8	ADDA	M1=M1+M0+D		
9	SUBA	M1=M1–M0+D		
10	Running average (running average every D points, M1 (n)= $\frac{1}{D}\sum_{k=n-\frac{D-1}{2}}^{n+\frac{D-1}{2}}$ M0 (k)) (D is odd number)			

Notes:

- When K is assumed to be 0 to 3, use the memory number 0 to 5, 8 or 9 for the memory called "integer", and use the memory number 6 or 7 for the memory called "real number".
- P0 and P1 are numeric constants or numeric variables from 0 to 500.
- D is a numeric constant or numeric variable. Its default is D=0.
- When K is 10, $(P0-\frac{D-1}{2}) \geq 0$ and $(P1+\frac{D-1}{2}) \leq 500$ must be satisfied.

SWLG subroutine

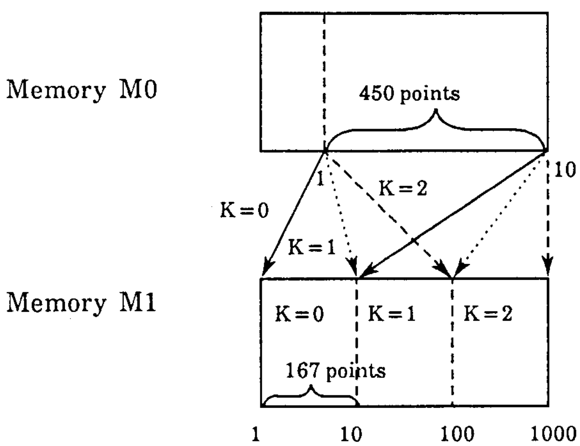
(1) Function

This subroutine arranges the data of the specified memory so that the frequency axis is LOG display and then transfers it.

(2) Format

```
CALL␣SWLG(K,MØ,M1)
```

- M1: Destination memory
- MØ: Source memory
- K: Function specification



K=0 : LEFT (1 to 10)
K=1 : CENTER (10 to 100)
K=2 : RIGHT (100 to 1000)

The memory M0 data is a measured value obtained by an ordinary (linear) sweep.
The frequency axis LOG for 3 decades can be displayed in memory M1 by sweeping three times by changing the frequency and by executing the SWLG subroutine three times.

Note: The M0 and M1 must be combined within the integer memories, or real M0 and M1 must be combined in the real number memories.

System Functions

The system functions can extract and calculate special points in the waveform data, with the waveform memory as the objective. Therefore, there is a function result value.

System function		Function
Maximum value	MAX(M, P0, P1 N)	Returns the maximum value between P0 to P1
Minimum value	MIN(M, P0, P1, N)	Returns the minimum value between P0 to P1
Frequency at specified measured value (1)	BNDL(M, P0, L, N)	Starts search from P0 and returns the frequency of the supecified measured value
Frequency at specified measured value (2)	BNDH(M, P0, L, N)	Starts search from P0 and returns the frequency at the supecified measured value
Frequency at specified measured value (3)	MESL(M, P0, L, N)	Starts search from P0 and returns the frequency of the supecified measured value
Frequency at specified measured value (4)	MESH(M, P0, L, N)	Starts search from P0 and returns the frequency of the supecified measured value
Ripple 1	RPL1(P0, P1, N [, R])	Obtains ripple 1 between P0 to P1
Ripple 2	RPL2(P0, P1, N [, R])	Obtains ripple 2 between P0 to P1
Ripple 3	RPL3(P0, P1, N [, R])	Obtains ripple 3 between P0 to P1
Peak 1	PEKL(M, P0, L, N [, R])	Starts search from P0 and returns peak value
Peak 2	PEKH(M, P0, L, N [, R])	Starts search from P0 and returns peak value
Pole 1	POLL(M, P0, L, N [, R])	Starts search from P0 and returns pole (dip) value
Pole 2	POLH(M, P0, L, N [, R])	Starts search from P0 and returns pole (dip) value
Inflection top value 1	PLRH(M, P0, N [, R])	Starts search from P0 and returns adjacent inflection maximum
Inflection top value 2	PLLH(M, P0, N [, R])	Starts search from P0 and returns adjacent inflection maximum
Inflection bottom value 1	PLRL(M, P0, N [, R])	Starts search from P0 and returns adjacent inflection minimum
Inflection bottom value 2	PLLL(M, P0, N [, R])	Starts search from P0 and returns adjacent inflection minimum

(Continued)

System function		Function
Frequency specified point	PFRQ(P0)	Returns frequency of P0 point
Total	SUM(P0, P1, N)	Returns total of the memory contents between P0 to P1
Addition search 1	PSML(M, P0, L, N)	Successively adds from P0 and returns a point with the specified value
Addition search 2	PSMH(M, P0, L, N)	Successively adds from P0 and returns a point with the specified value
Decision 1	DPOS(M, P0, P1, N1, N2)	Compares and decides the size of the memory contents
Decision 2	DNEG(M, P0, P1, N1, N2)	Compares and decides the size of the memory contents

Notes:

- Since the waveform memory is the objective of the system functions, the input values (P0 and P1) to each function are specified as points on all the waveform memories.
- P0, P1, L, N and R are input parameters indicated by a numeric constant or numeric variable.
- M is an output parameter indicated by a variable.
- N, N1 and N2 are parameter which specify the waveform memory. It is a numeric constant or numeric variable.

N, N1, N2	Memory	System variable name	Type
0	Measurement memory TRACE-A	XMA ( )	Integer
1	Measurement memory TRACE-B	XMB ( )	Integer
2	Submemory a	SMA ( )	Integer
3	Submemory b	SMB ( )	Integer
4	Image memory a	IMA ( )	Integer
5	Image memory b	IMB ( )	Integer
6	Real number memory a	RMA ( )	Real number
7	Real number memory b	RMB ( )	Real number
8	Measurement memory TRACE-TIME	XMT ( )	Integer
9	Measurement memory TRACE-BG	XMG ( )	Integer
10	Sub-memory t	SMG ( )	Integer

- [,R] can be omitted. When omitted, R is assumed to be 0.
- P0 and P1 specify the points in the waveform memory. Their setting range is 0 to 1001.
- P0 and P1 used in the system functions always specify the points in the measurement memories.

MAX function

(1) Function

This function obtains the maximum value in the specified measurement memory area and the measurement point at the maximum value.

(2) Format

```
MAX(M,PØ,P1,N)
```

- N: Measurement memory specification
- PØ,P1: Area Specification (PØ<P1)
- M: Measurement point at maximum value
- MAX: Maximum value

Note: If there is more than one point with the same maximum value, the first point of the maximum value is stored in M.

(3) Program example: Obtains maximum level in measurement memory TRACE-A.

```
1Ø  REM "MAX(M,PØ,P1,N)"
2Ø  GMAX=MAX(M,Ø,5ØØ,Ø)
3Ø  GMAX=GMAX*Ø.Ø1
4Ø  PRINT "Maximum Level=",GMAX,"dBm"
5Ø  STOP
Maximum Level=–2Ø.45dBm
```

MIN function

(1) Function

This function obtain the minimum value in the specified measurement memory area and the measurement point at the minimum value.

(2) Format

```
MIN(M,PØ,P1,N)
```

Measurement memory specification
Area Specification (PØ<P1)
Measurement point at minimum value
Minimum value

Note: If there is more than one point with the same minimum value, the first minimum value point is stored in M.

(3) Program example: Obtains minimum level in measurement memory TRACE-B.

```
1Ø  GMIN=MIN(M,Ø,5ØØ,1)
2Ø  GMIN=GMIN*Ø.Ø1
3Ø  PRINT "Min Level=",GMIN,"dBm at",M
4Ø  STOP
```

BNDL, BNDH, MESL, and MESH functions

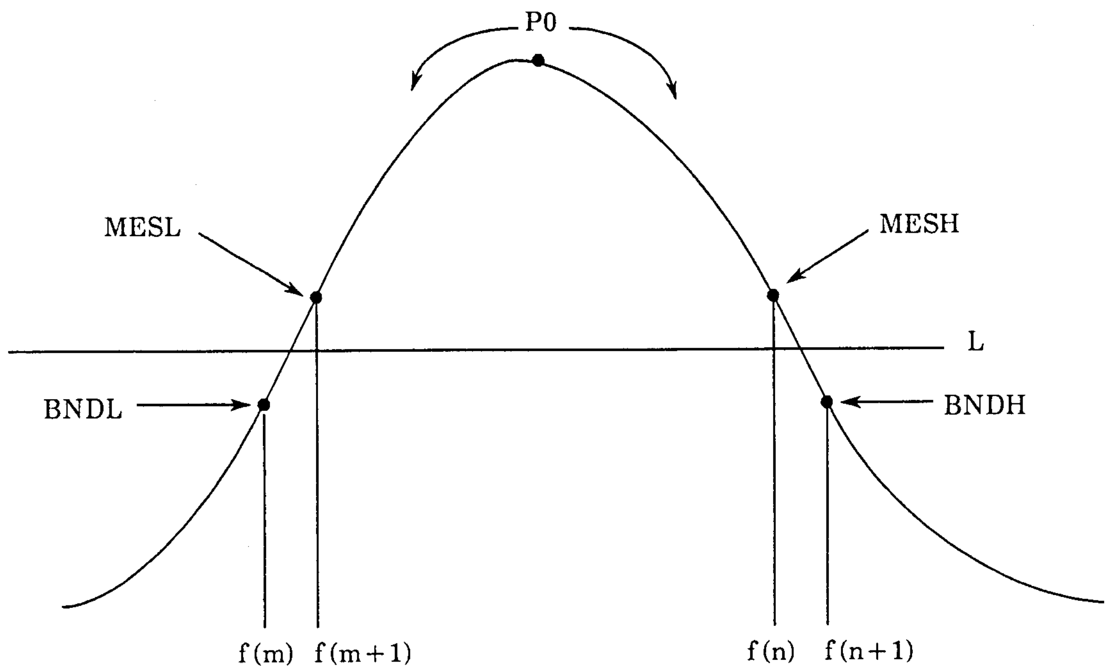
(1) Function

These functions obtain the frequency at the specified measured value by searching from a starting point in the specified memory.

(2) Format

```
BNDL(M,PØ,L,N)
BNDH(M,PØ,L,N)
MESL(M,PØ,L,N)
MESH(M,PØ,L,N)
  |    |  | | |
  |    |  | | └── Measurement memory specification
  |    |  | └──── Specified measured value
  |    |  └────── Search starting point
  |    └───────── Measurement point at specified measured value
  └────────────── Frequency (or time) at specified measured value
```

- When N is specified to 0, 2, 4, 6, 7
  Find the frequency of the specified measurement value from the TRACE-A setting frequency.
- When N is specified to 1, 3, 5, 7
  Find the frequency of the specified measurement value from the TRACE-B setting frequency.
- When N is specified to 8, 10
  Find the time of the specified measurement value from the TRACE-TIME setting time.
- When N is specified to 9
  Find the frequency of the specified measurement value from the TRACE-BG setting time.



Note: If there is no specified measured value in BNDL and MESL, M is assumed to be 0; in BNDH and MESH, M is assumed to be 1001.

(3) Program example: Obtains bandwidth at level of –20 dBm in A channel memory, searching from center.

```
1Ø   L=-2ØØØ ........... indicates –20 dBm
2Ø   FL=BNDL(ML,25Ø,L,Ø)
3Ø   FH=BNDH(MH,25Ø,L,Ø)
4Ø   BW=(FH-FL)/1ØØØ
5Ø   PRINT "BW=",BW,"KHz"
6Ø   STOP
```

RPL1 and RPL2 functions

(1) Function

These functions obtain ripple 1, and 2 in the specified memory area.

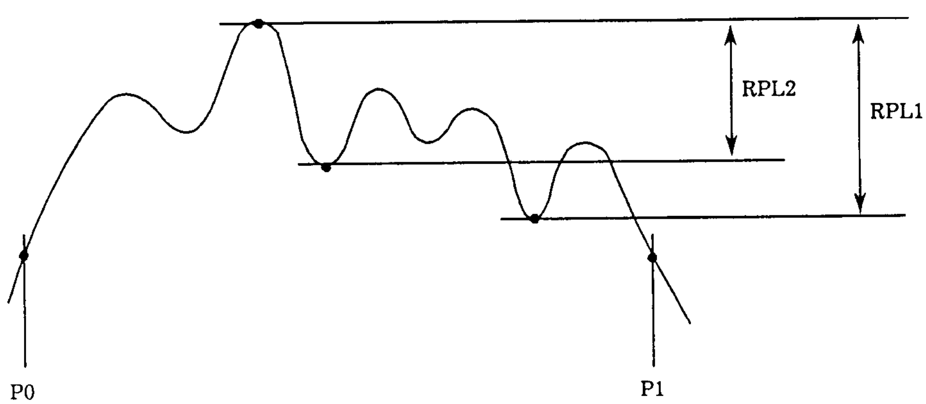
Ripple 1: This is the difference between the maximum value of the inflection top value and the minimum value of the inflection bottom value.

Ripple 2: This is the maximum difference between the adjacent inflection top and bottom values.

(2) Format

```
RPL1(PØ,P1,N[,R])
RPL2(PØ,P1,N[,R])
```

- R: Resolution of difference between adjacent inflection top and bottom values
- N: Measurement memory specification
- PØ,P1: Area specifications (PØ<P1)
- RPL2: Ripple



Notes:

- If the difference between the adjacent inflection top and bottom values is smaller than R, the ripple is not obtained.
- N which specifies the measured memory must be from 0 to 5, 8 or 9. (No real number memory can be used.)

(3) Program example: Obtains Ripple 1 between the measurement points 100 and 300 in measurement memory TRACE-A, where resolution is 0.2 dB.

```
1Ø  RP=RPL1(1ØØ,3ØØ,Ø,2Ø,) ........... R=20 when resolution is 0.2 dB
2Ø  RP=RP/1ØØ
3Ø  PRINT "RPL1=",RP,"dB"
4Ø  STOP
```

RPL3 function

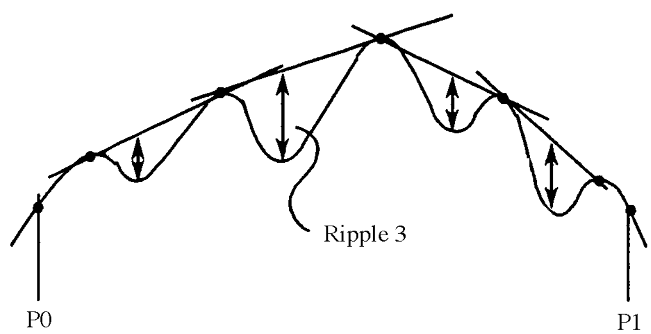
(1) Function

This function obtains the maximum difference between the adjacent tangent at the inflection top and inflection bottom value (ripple 3) in the specified memory area as shown a figure below.

(2) Format

```
RPL3(PØ,P1,N[,R])
```

- R: Resolution of difference between adjacent inflection top and bottom values
- N: Measurement memory specification
- PØ,P1: Area psecification (PØ<P1)
- RPL3: Ripple



Notes:

- If the difference between the adjacent inflection top and bottom values is smaller than R, the ripple is not obtained.
- N which specifies the measured memory must be from 0 to 5, 8 or 9. (No real number memory can be used.)

(3) Program example:

Obtains Ripple 3 between the measurement points 50 and 450 in the measurement memory TRACE-B, where resolution is 0.1 dB.

```
1Ø  RP=RPL3(5Ø,45Ø,1,1Ø,)
2Ø  RP=RP/1ØØ
3Ø  PRINT "RPL3=",RP,"dB"
4Ø  STOP
```

PEKL and PEKH functions

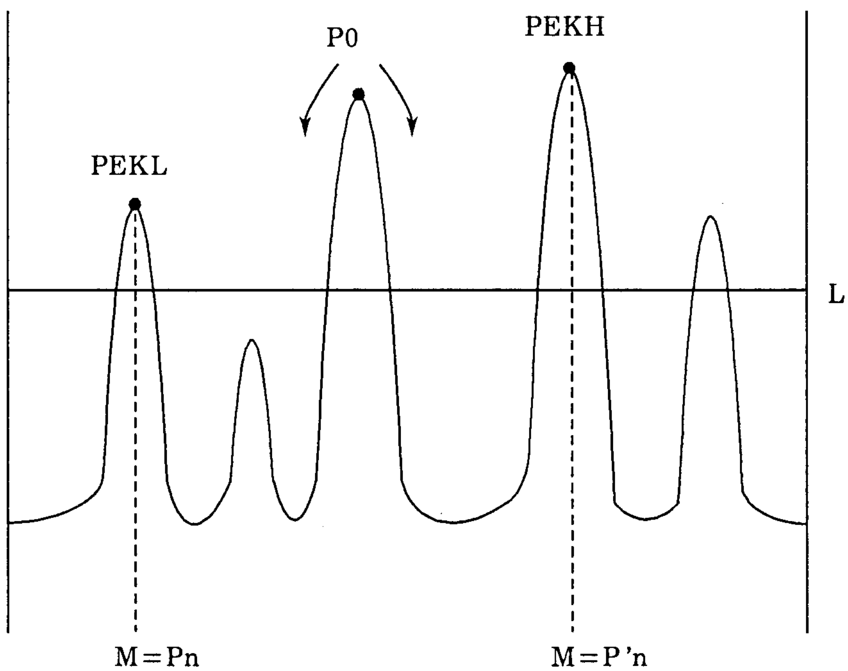
(1) Function

These functions find the first peak and its measured point, which is larger than the specified measured value in the measurement area, by searching from a starting point in the specified memory.

(2) Format

```
PEKL(M,PØ,L,N[,R])
PEKH(M,PØ,L,N[,R])
```

- R: Resolution for difference between adjacent inflection top and bottom value
- N: Memory specification
- L: Specified measured value
- PØ: Search starting point
- M: Peak measurement point
- PEKL/PEKH: Peak level



Notes:

- If the peak cannot be found with the PEKL function, M is assumed to be 0, and the measured value at point 0 is PEKL.
- If the peak cannot be found with the PEKH function, M is assumed to be 500, and the measured value at point 1001 is PEKH.
- N which specifies the measured memory must be from 0 to 5, 8 or 9. (The real number memory cannot be used.)
- If the difference between adjacent inflection top and bottom values is smaller than R, the inflection top is not the peak.

(3) Program example: Obtains peak level higher than –50 dBm searched left of the measurement point 200 in measurement memory TRACE-A, where resolution is 2 dB.

```
1Ø  PLEV=PEKL(M,2ØØ,-5ØØØ,Ø,2ØØ)
2Ø  PLEV=PLEV/1ØØ
3Ø  PRINT "Peak Level=",PLEV,"dBm at",M
4Ø  STOP
```

POLL and POLH functions

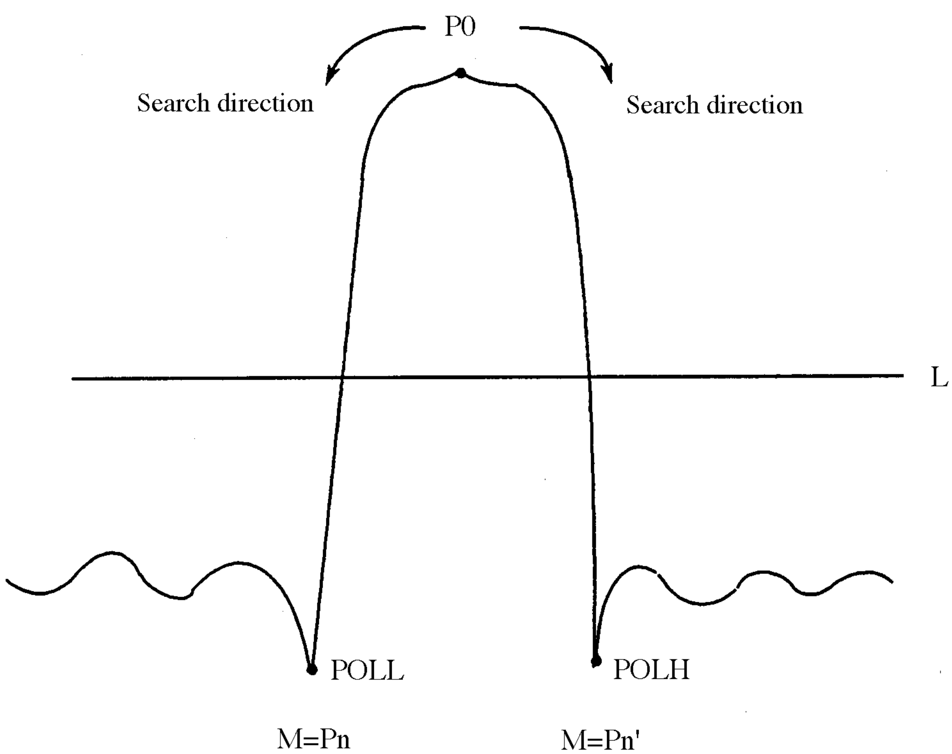
(1) Function

These functions obtain the pole and its measurement point, which is smaller than the specified measured value in the measurement area, by searching from a starting point in the specified memory.

(2) Format

```
POLL(M,PØ,L,N[,R])
POLH(M,PØ,L,N[,R])
```

- R: Resolution for difference between adjacent inflection top and bottom values
- N: Measurement memory specification
- L: Specified measured value
- PØ: Search starting point
- M: Measurement point at pole
- POLL / POLH: Pole level



Notes:

- If pole cannot be obtained in POLL function, M is assumed to be 0, and the measured value at point 0 is POLL.
- If pole cannot be obtained in POLH function, M is assumed to be 1001, and the measured value at point 500 is POLH.
- N which specifies the measured memory must be from 0 to 7, 8 or 9. (No real number memory can be used.)
- If the difference between adjacent inflection top and bottom values is smaller than R, the inflection top is not the pole.

(3) Program example: Obtains pole level lower than –60 dBm searched left of the measurement point 250 in measurement memory TRACE-A, where resolution is 1 dB.

```
1Ø  PL=POLL(M,25Ø,-6ØØØ,Ø,1ØØ)
2Ø  PL=PL/1ØØ
3Ø  PRINT "Poll Level=",PL,"dBm at",M
4Ø  STOP
```

PLRH, PLLH, PLRL and PLLL functions

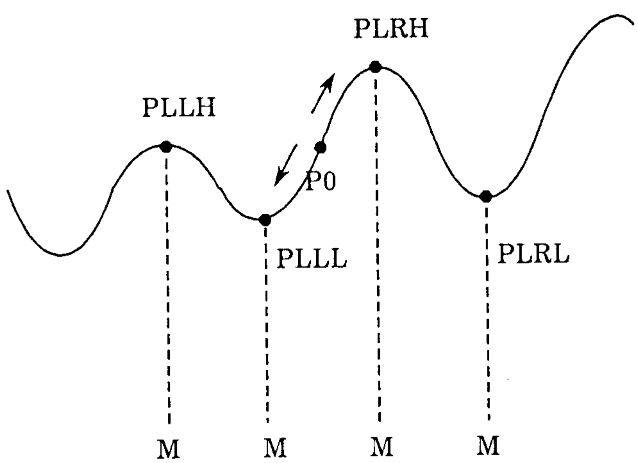
(1) Function

These functions obtain the first inflection top and bottom values and their measurement points by searching from a starting point in the specified memory.

(2) Format

```
PLRH(M,PØ,N[,R])
PLLH(M,PØ,N[,R])
PLRL(M,PØ,N[,R])
PLLL(M,PØ,N[,R])
```

- R: Resolution for difference between adjacent inflection top and bottom values
- N: Measurement memory specification
- PØ: Search starting point
- M: Measurement points of the inflection top and bottom values
- PLRH, PLLH, PLRL, PLLL: Inflection top and bottom values



Notes:

- If the difference between the adjacent inflection top and bottom values is smaller than R, the two points are not the inflection points. If R is omitted, it is assumed to be 0.
- If there is no inflection top and bottom point, M is assumed to be 0 at PLLH and PLLL and M is assumed to be 1001 at PLRH and PLRL; the measured value at point 0 is PLLH and PLLL and that at point 1001 is PLRH and PLRL.
- N specified by measured memory must be from 0 to 7, 8 or 9. (No real number memory can be used.)

(3) Program example: Obtains inflection top level searched right of the measurement point 200 in measurement memory TRACE-B, where resolution is 3 dB.

```
10  PL=PLRH(M,250,1,300)
20  PL=PL/100
30  PRINT "Peak Level=",PL,"dBm at",M
40  STOP
```

PFRQ function

(1) Function

This function finds the frequency of the specified point or time in the memory.

(2) Format

```
PFRQ(PØ)
     └──── Specified point
```

Notes:

- When the effective trace setting on the CRT is frequency domain (TRACE-A, B, BG), the frequency is output; and when it is time domain (TRACE-TIME) the time is output.
- Frequency is output in 1 Hz units and time is output in 1µs units.
- This function finds frequency values by the following equations:

$$\text{Frequency=start frequency+}\frac{\text{P0}}{500}*\text{(frequency span)}$$

(3) Program example:

Obtains maximum level between the measurement points 100 and 300 and frequency at that point in the measurement memory TRACE-A.

```
1Ø  GMAX=MAX(M,1ØØ,3ØØ,Ø)
2Ø  FR=PFRQ(M)
3Ø  GMAX=GMAX/1ØØ
4Ø  FR=FR/1E6
5Ø  PRINT "Peak Freq=",FR,"MHz"
6Ø  PRINT "Peak Level=",GMAX,"dBm"
7Ø  STOP
```

SUM function

(1) Function

This function finds the sum of the memory contents of a certain interval in the specified memory.

(2) Format

```
SUM(PØ,P1,N)
 │   └─┬─┘ └── Memory specification
 │     └────── Area Specification (PØ<P1)
 └──────────── Sum
```

$$SUM = \sum_{k=P0}^{P1} L(k)$$

(3) Program example:

Obtains average value between the measurement points 240 and 260 (21 points) in measurement memory TRACE-A.

```
1Ø  S=SUM(24Ø,26Ø,Ø)
2Ø  AV=S/21/1ØØ
3Ø  PRINT "Average=",AV:F7.2,"dBm"
4Ø  STOP
```

Note: When the measurement memory contains invalid data (points with marker level displayed as ***), that data is assumed to be –30000 (=–300.00 dBm) and calculation is performed.

PSML and PSMH functions

(1) Function

This function finds the point where the sum equals or exceeds the specified value while adding the memory contents sequentially by searching from a starting point in the specified memory.
(For example, this is used to measure the occupied bandwidth)

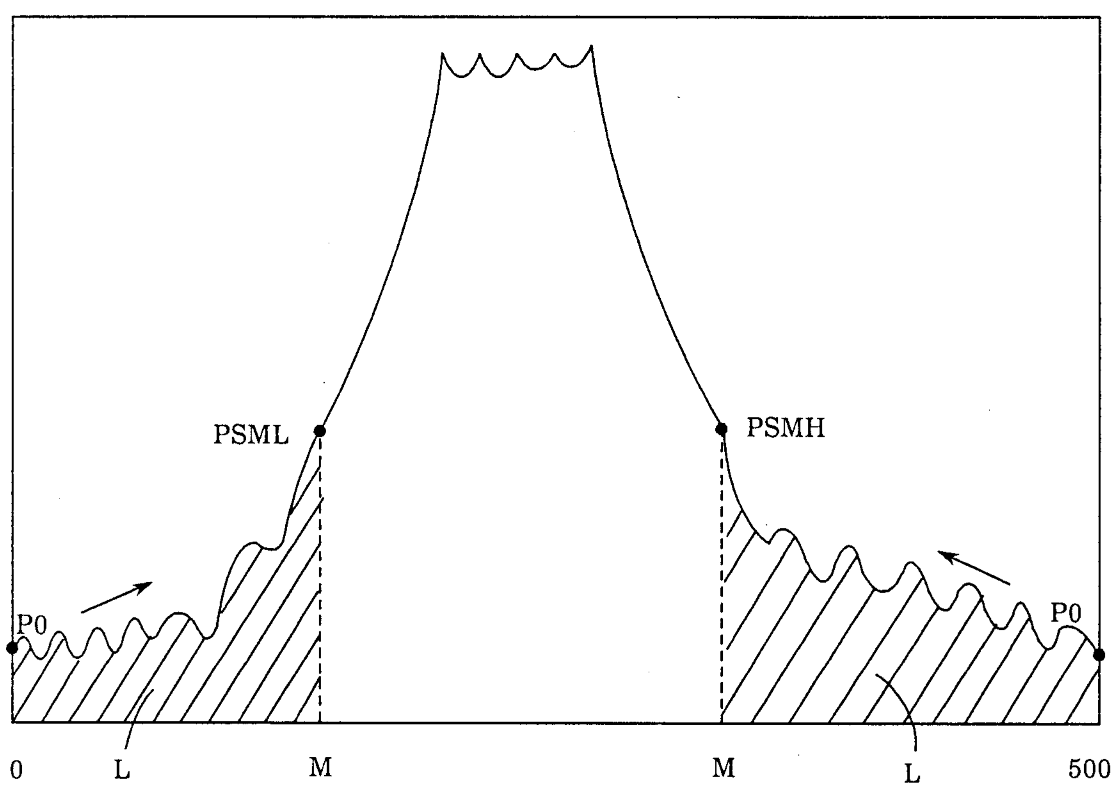
Finding method of the frequency or time depends on the specified waveform memory number.

See Section 5, "BNDL, BNDH, MESL and MESH functions" for details.

(2) Format

```
PSML(M,PØ,L,N)
PSMH(M,PØ,L,N)
```

- N: Memory specification
- L: Specified value
- PØ: Search starting point
- M: Measurement point
- PSML / PSMH: Frequency at measurement point (or time)



PSML: Finds the minimum value of M that satisfies

$$L \leq \sum_{k=P0}^{M} L(k)$$

PSMH: Finds the maximum value of M that satisfies

$$L \leq \sum_{k=M}^{P0} L(k)$$

(3) Program example: Converts the measurement data in measurement memory TRACE-A to real value of mW unit, obtains sum of total data and frequency of the point, where sum equals 0.5% of the total sum adding the memory contents by searching from left end (address 0).

```
1Ø  CALL CONV(2,Ø,6,Ø,5ØØ)
2Ø  T=SUM(Ø,5ØØ,6)
3Ø  L=T*Ø.ØØ5
4Ø  FR=PSML(M,Ø,L,6)
5Ø  FR=FR/1E6
6Ø  PRINT "Point=",M
7Ø  PRINT "Freq=",FR,"MHz"
8Ø  STOP
```

DPOS and DNEG functions

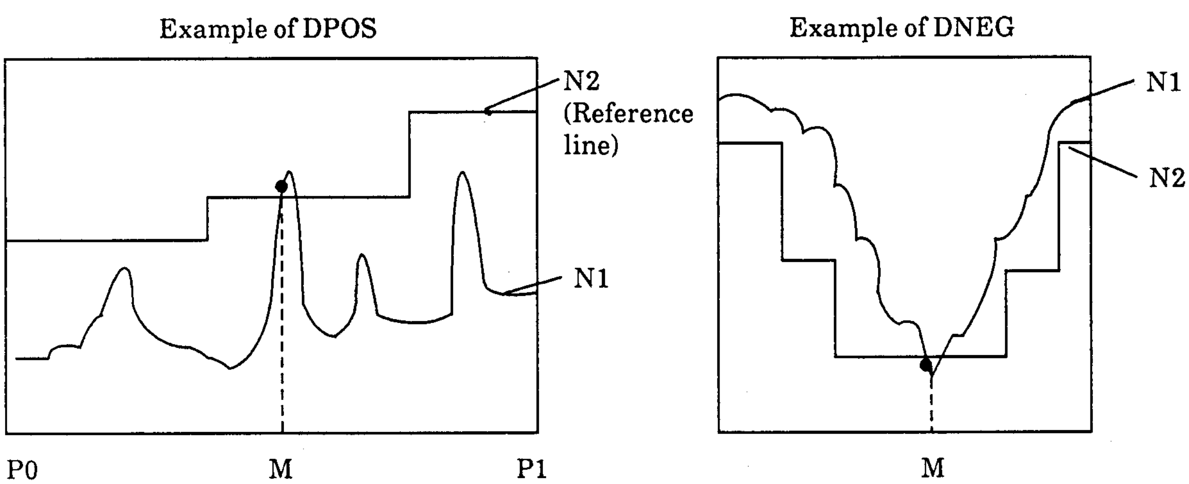
(1) Function

These functions compare the contents of two memories by address. If a value in one memory is larger (or smaller) that the other even if at only one point, the function value is assumed to be 1. Otherwise, 0 is output. (For example, this is used to judge GO/NOGO for the standard.)

(2) Format

```
DPOS(M,PØ,P1,N1,N2)
DNEG(M,PØ,P1,N1,N2)
```

- N1,N2 — Memory specification
- PØ,P1 — Area specification
- M — Measurement point (the leftmost point exceeding the condition)
- DNEG — Results Ø: GO
  1 : NOGO



DPOS: 1 when there is a N1>N2 point; 0 when there is no N1>N2 point

DNEG: 1 when there is a N1<N2 point; 0 when there is no N1<N2 point

(3) Program example:

Compares the measurement data in measurement memory TRACE-A with measurement data in measurement memory TRACE-B and displays GO or NOGO.

```
1Ø  X=DPOS(M,Ø,5ØØ,Ø,1)
2Ø  IF X=Ø PRINT "GO"
3Ø  IF X=1 PRINT "NO GO"
4Ø  STOP
```

SECTION 6

REMOTE CONTROL COMMANDS USED WITH PTA PROGRAM/LIBRARY

TABLE OF CONTENTS

SECTION 6
REMOTE CONTROL COMMANDS USED WITH PTA PROGRAM/LIBRARY

Outline

Remote control commands to control the main frame side, using PUT and WRITE 1000 texts in a PTA program/library, are sent. Also, using GET, COM and READ 1000 texts, measurement parameters and measurement results of the main frame side are read out. Remote control commands available here include all control and inquiry commands defined on the MS2665C/67C/68C main frame side. In addition, there are also remote control commands specially prepared for PTA programs/libraries.

PTA Dedicated Remote Control Commands

When setting or reading parameters of a measuring instrument on the PTA main frame side, messages in the remote control command format are sent using the WRITE 1000 or READ 1000 statement.

In PTA, besides the remote control commands of MS2665C/67C/68C, the following messages can be sent out.

Function	Message		
Port Switching	Control	PORT␣1	; Selects RS-232C as the PTA control port.
		PORT␣2	; Selects GPIB as the PTA control port.
		PORT␣3	; Selects the parallel (centronics) as the PTA controller port.
	Request	PORT?	; Requests the PTA control port.
Event Occurrence DELAY (Clock 1)	Control	EDLY␣t	; Sets the DELAY time an event interrupt will occur. t — DELAY time: 1 seconds up to 1 hour (in 1 s step)
Event Occurrence TIME (Clock 2)	Control	ETIM␣t1,t2,t3	; Sets the time an event interrupt will occur t3 — Seconds: Up to 59 seconds t2 — Minutes: Up to 59 minutes t1 — Hours: Up to 23 hours
Event Occurrence CYCLE (Clock 3)	Control	ECYC␣t	; Sets the cycles an event interrupt will occur. t — Cycle: 1 seconds up to 1 hour (in 0.1 s steps)

- For details on the WRITE 1000 and READ 1000 statements, see Section 4, "Setting measurement parameters (PUT and WRITE 1000 statements)" and "Measurement parameter/data read (GET, COM and READ 1000 statements)".
- For details on event interrupts, see Section 4, "ENABLE EVENT statement".
- The control port (for the WRITE, READ, LISTG statements and other GPIB statements supported by the PTA) is the port selected by the PORT command except when these statements are executed with a direct port specification.
  In the initial state, the GPIB1 port is selected as the PTA control port.
- Ports specified by the port switching command are not initialized by PTA→OFF.

SECTION 7

EXTERNAL INTERFACE IN PTA

TABLE OF CONTENTS

SECTION 7 EXTERNAL INTERFACE IN PTA

Outline

MS2665C/67C/68C provides an RS-232C interface and a GPIB interface as standard, and a parallel (centronics) interface (option 10) is optionally available. These external interfaces can be controlled from PTA.

Selection of Controlled Interface Port from PTA

An interface port controlled from PTA is selected by the "connection port for peripheral devices (Connect to Peripheral)" of the Interface menu.

(1) Press [SHIFT] + [.:Interface] keys.

(2) Press the F6 key "connection port for peripheral devices (Connect to Peripheral)" several times to display candidate interface ports for selection.

If the interface port to be controlled from PTA has been set as the "connection port for the external controller (Connect to Controller)" or the "connection port for the printer/plotter (Connect to Printer/Plotter)", first switch the selection to another port or make it "no connection (NONE)" and then operate the F6 key "connection port for peripheral devices (Connect to Peripheral)".

Also, using the PORT remote command or CALL IFC subroutine, it is possible to make the external interface port forcibly controllable from PTA.

- PORT_1: This command forcibly sets the connection port for external devices as the RS-232C interface.
- PORT_2: This command forcibly sets the connection port for external devices as the GPIB interface.
- PORT_3: This command forcibly sets the connection port for external devices as the parallel (centronics) interface.
- CALL IFC: This command forcibly sets the connection port for external devices as the GPIB interface.

RS-232C Functions in PTA

(1) Program listing

The LISTG command lists programs from the RS-232C port to an external printer.

(2) Data sending

The WRITE statement sends data to a device connected to the RS-232C.

```
WRITE␣M,Variable[:Format][,Variable[:Format]····]
```

Output data (A character constants available.)

External device address (numeric constant or variable used.)

(3) Data receiving

The READ statement receives data from a device connected to the RS-232C.

```
READ␣M,Variable [,Variable····]
```

Received data is input in the variable.

External device address (numeric constant or variable used.)

(4) Time-out

The time-out time is input as five seconds (initial value).
Use the following GPIB command to change the time-out time:

`TOUT␣t` t=0 to 255 seconds (second unit)

If t=0 is specified, no time-out is set.

(5) Terminating Codes for READ/WRITE Statements

The following terminating codes are used for the RS-232C port.

Send terminators

<Port> command	Terminator code
WRITE LISTG	Either CR+LF or LF (Comply with TRM command)

Receive terminators

<Port> command	Terminator code
READ	LF or CR + LF

GPIB Functions in PTA

Function as controller

When the GPIB interface port is set as the "connection port for peripheral devices (Connect to Peripheral)", GPIB functions as a controller.

(1) Program listing

Lists programs to an external printer by using the LISTG command through the current GPIB port.

(2) IFC sending

Sends the "Interface Clear" to the device on the GPIB by using the CALL␣IFC statement.

(3) Controller right allocation

Allocates controller right to the device with the address specified by M by using the CALL␣TCT (M) statement .

(4) Data sending

Sends the data to the device on the GPIB by using the WRITE statement

```
WRITE␣M,Variable[:Format][,Variable[:Format]…]
```

Output data (A character constant is possible.)

Address of external device (A numeric constant or numeric variable is used.)

NOTES

When M is 1000, the functions of the MS2665C/67C/68C main frame are set. Also, this operations are performed in either the controller or device mode at this time.

(5) Data reception

Receives the data from the device on the GPIB by using the READ statement

```
READ␣M,Variable [,Variable…]
```

Received data is input in variable.

Address of external device (A numeric constant or numeric variable is used.)

NOTES

When the specified GPIB port is the device port, WRITE and READ statements access the dual-port memory.

NOTES

When one- or two-digit value (e.g.,5 or 17) is specified for an address, the value indicates the address of the device connected to the port specified by the PORT command of the GPIB command (Indirect Port Specification). When a three-digit value (e.g.,105 or 217) is specified, the high-order digit indicates the port number, and two low-order digits indicate the address of the device connected to the port indicated by the above port number. (Direct Port Specification).

The two lower digits of an address at indirect or direct port specification have no meaning in RS-232C. However, these digits should still be specified for form's sake.

Example:

`WRITE␣5,"ABC"` .... Data is sent to address 5 through the current port (indirect port specification).

`WRITE␣1Ø5,"ABC"` .... Data is sent to address 5 through the specified port No.1 (RS-232C) (direct port specification).

`READ␣217,A$` .... Data is input from address 17 through the specified port No.2 (GPIB) (direct port specification).

These address specifications are effective for the WRITE, BWRITE, WWRITE, READ, BREAD, WREAD and LISTG statements.

The relationship (between the port specification command and controller port) is as follows:

	Indirect port specification	Direct port specification	
	`WRITE 5`	`WRITE 1Ø5`	`WRITE 2Ø5`
At power-on or after "PORT␣1" execution	*1 The RS-232C port is the controller port.	*1 The RS-232C port is the controller port.	*2 The GPIB port is the controller port.
After "PORT␣2" execution	*2 The GPIB port is the controller port.	*1 The RS-232C port is the controller port.	*2 The GPIB port is the controller port.

*1 Address specification in the RS-232C has no meaning. However, the address should still be specified for form's sake.

*2 If the GPIB port is not the controller port due to the CALL IFC statement, it controls the dual port memory. In this case, the LISTG statement becomes ineffective.

When the specified port is a device port, data is written to and read from the dual port memory. In this case, the BWRITE, WWRITE, BREAD, WREAD, and LISTG statements cannot be used.

(6) Time out

The time-out value is 30 s (initial value).

The following GPIB command is used for change of thime-out value.

GTOUT␣t　　t=0 to 225 s (in 1 s steps)

When t=0 is specified, no time-out is set.

(7) Terminating Codes for READ/WRITE Statements

The following terminating codes are used for the GPIB ports.

Talker (send) terminators

<Port> command	Terminator code
<GPIB> WRITE LISTG	Depends on TRM command. eiter CR + LF or LF

Note:
The TRM command shown below is a GPIB command.

TRM␣1 (CR + LF)
TRM␣Ø (LF only)
Initial value : LF only

Listener (receive) terminators

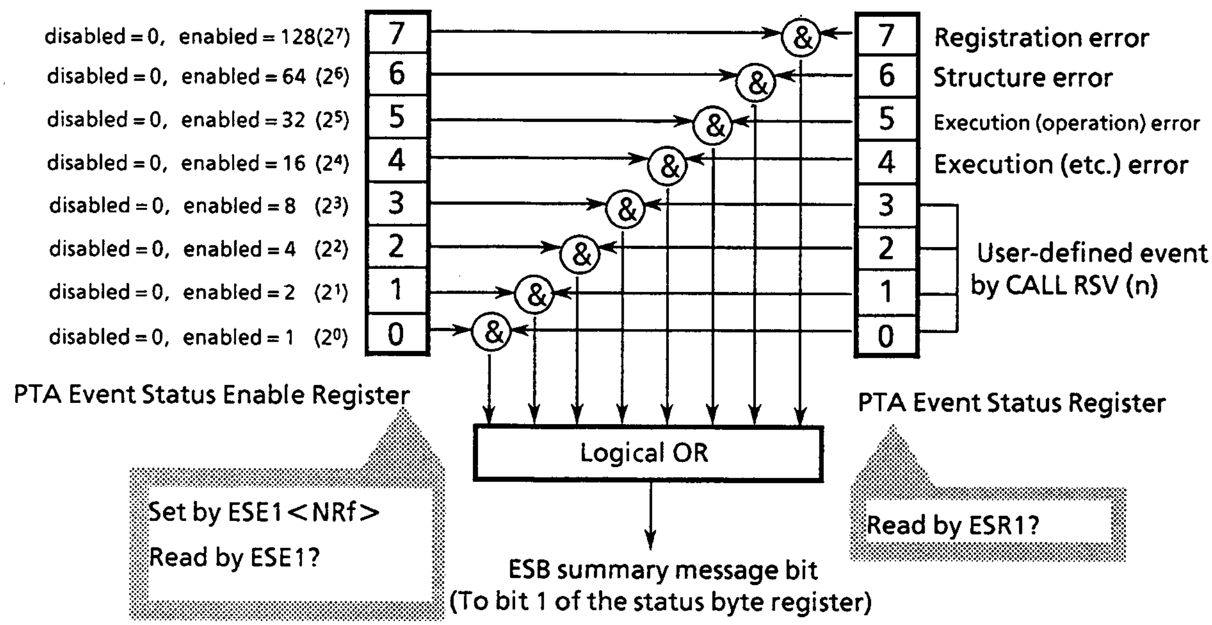
<Port> command	Terminator code
<GPIB> READ	LF or CR + LF

Function as device

When the GPIB interface port is set as the "connection port for the external controller (Connect to Controller)", GPIB functions as a device.

(1) Service request sending

Sends a service request command to an external controller by using the CALL␣RSV (M) statement.



Bit	Event name	Description
7	Registration error	Error at program registration
6	Structure error	Error on program structure
5	Execution (operation) error	Error at operation on program execution
4	Execution (etc.) error	Error at other than program operation
3	(User-defined event)	(User defined by CALL RSV (n))
2	(User-defined event)	(User defined by CALL RSV (n))
1	(User-defined event)	(User defined by CALL REV (n))
0	(User-defined event)	(User defined by CALL RSV (n))

Functions of Parallel (centronics) in PTA

(1) Program listing

The LISTG command lists programs from the parallel (centronics) port to the external printer.

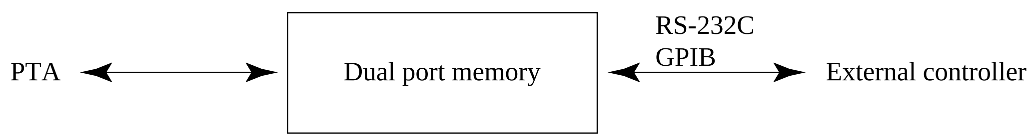
Dual Port Memory

(1) Application and configuration

The dual port memory is built in PTA, and data can be freely written and read from PTA and the external controller.

Data and measurement results obtained in the PTA program/library are outputted to the external controller through this memory, and used for performing communication between PTA and external controller.

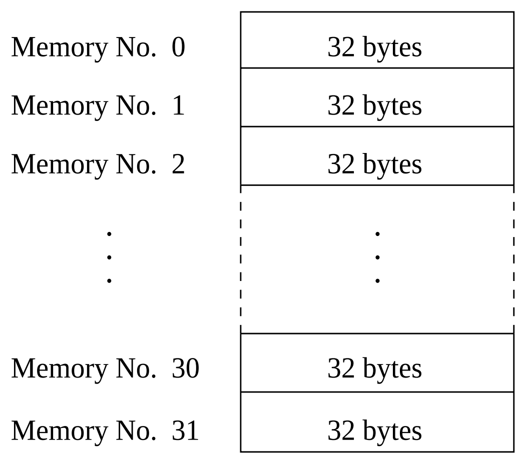
The external controller writes to and reads from the dual port memory through the interface set as the "connection port for the external controller (Connect to Controller)".



The dual port memory consists of thirty-two 32-byte memories. The memories are accessed by specifying the memory number.

Memory numbers from 0 to 31 can be specified.

Dual port memory configuration



(2) Writing data to dual port memory

Format

- Writing from PTA

```
WDPM memory number, write data   or
PUT(or WRITE 1ØØØ) " PMY memory number, write data"
```

- Writing from external controller

```
" PMY memory number, write data"
```

- When writing data to the dual port memory, be sure to specify the memory number. Data is written sequentially, beginning from the first byte of the specified memory number.
- A 1-byte termination code (LF) is added at the end of the write data.
- When the write data size exceeds 32 bytes, it can be written to the next memory. When the write data size is exactly 32 bytes, the termination code is stored at the beginning of the next memory. However, when data has been written up to the last byte of the last memory number, the termination code is not added.
- When writing past the last byte of the last memory number is attempted, an error is generated and writing is not performed. In this case, the previously written data is retained.
- Data is always stored in memory as ASCII data. When data is written from the PTA, its storage size differs as follows, depending on the type of data:

1) Character constant/variable

- Written as 1 byte/1 character ASCII data.
- When unformatted character variable data is written, (number of bytes of array size)+(1 byte: space code) is written. The termination code is written at the end.
- When upper case formatted character variables are used, a 1-byte space code is written at the end of the data. The termination code is written at the end.
- When character variables are used, the number of characters in "   " are written. The termination code is written at the end.

2) Numeric variable

- Numerics are converted to character strings (ASCII data) and data of that size is written.
  The minus sign and decimal point require one byte each.
  The termination code is written last.

3) Bit variable

- The 0/1 numeric of each bit is converted to a character string (ASCII data) and data of that size is written as 1 byte/1 bit.
- The storage format when the data is formatted/unformatted is the same as when character variables are used.
- The BWRITE and WWRITE statements cannot be used.

Examples:

- Writing from PTA

  `WDPM Ø, "MEASEND"` : Write "MEASEND" to Memory No. 0.

- Writing from external controller

  `"PMY Ø, MEASSTART"` : Write "MEASSTART" to memory No.0.

Notes:

- The WDPM statement is a dedicated statement for writing data to dual port memory.
- The PUT or WRITE 1000 statement is mainly used to set measurement parameters of the main frame. However, messages in the same format as setting from the external controller can be written using these commands by sending messages in the remote control command format from PTA.

(3) Reading data from dual port memory

Format

- Reading from PTA

```
RDPM memory number, input variable[,input variable..]  or
PUT(or WRITE 1ØØØ) "PMY? read start memory number, number of memories"
+READ 1ØØØ, input variable[,input variable]
```

- Reading from external controller

```
"PMY? read start memory number, number of memories" + read command
```

- When reading data from the dual port memory, be sure to specify the memory number. Everything up to the termination code (LF) is, as a rule, output as one data item.

  However, when dual port memory was read up to the last byte of the last memory number, the data is assumed to end at that point.

- When data was written over multiple memories and is read by specifying an intermediate memory number, the intermediate data is read.

- As a rule, when data is read from the PTA, the data up to the termination code is read. However, if the data contains commas (" , "), the commas are assumed to be delimiters and the data up to the front of the comma is stored in the input variable. Therefore, in this case, multiple input variables must be specified.

  When the number of delimited data and the number of input variables is different, a write error (when the number of input variables is large) may be generated, or the output data may remain inside (when the number of input variables is small).

  To avoid a comma being considered a data delimiter, store the data up to the termination code in one input variable by specifying " ; " at the end of the statement.

  In this case, only one input variable can be specified.

- When data is read from an external controller and when data is read from the PTA with the PUT or WRITE 1000 statement, use the "PMY?" command. The "PMY?" command can specify the read start memory number and the number of memories to be read. In this case, the data from the beginning to the termination code of each memory number is delimited into the specified number of memories by commas and is output.

- When the data in the dual port memory is assigned to input variables, it may not be possible to assign the data to an input variable type different from the assignment data. In this case, a read error is generated.

- The BREAD and WREAD statements cannot be used.

Examples:

- Reading from PTA

  `RDPM Ø, A$` : Read data from Memory No. 0 and store it in character variable A$.

- Reading from external controller

  `"PMY? Ø, 3"` : Issue a memory data output request for Nos. 0 to 3 (memory Nos. 0, 1, 2).

Notes:

- The RDPM is a dedicated statement for reading data from dual port memory.

(4) Details of write/read the dual-port memory

Control command from external controller

```
"PMY␣Ø,ABC"
"PMY␣1,123"
"PMY␣2,XYZ"
```

Contents of dual-port Memory

Memory 0	ABC (LF)
Memory 1	123 (LF)
Memory 2	XYZ (LF)

After executing statements shown on the above left, the contents of the dual-port memory are as shown on the above right.

When these data are read using "PMY?" command, the following contents are stored in variables A$, B$, and C$, respectively.

```
PUT␣"PMY?␣Ø,3"
READ␣1ØØØ,A$,B$,C$
```

Response message	ABC	,	123	,	XYZ	terminator

- Comma <,> in dual-port memory

  The output data for PMY? is assumed to be everything from the beginning to the <termination> code of the specified memory number. The output data includes the memory contents up to (but not including) the terminator. If a comma <,> is included in the contents, it indicates the presence of output data.

  In contrast, data in the READ statements for the PTA and controller are separated by commas and sequentially assigned to data variables. Therefore, the number of output variables generated by the PMY? command may be different from the number of variables required for the corresponding statement.

Contents of dual-port Memory

Memory 0	ABC, DEF (LF)
Memory 1	XYZ (LF)

Execute the statements shown below to read the contents of the dual-port memory at addresses 0 and 1.

```
PUT␣"PMY?␣Ø,2"
READ␣1ØØØ,A$,B$
```

The ABC represents data for variable A$ and the DEF represents data for variable B$. The contents of the memory 0 are separated by a comma (,). This comma separates the data into two data values. Consequently, the XYZ data in the memory 1 is not read. Therefore, the number of input variables in the READ statement must be set to three.

SECTION 8

PTA ERROR MESSAGES

TABLE OF CONTENTS

SECTION 8
PTA ERROR MESSAGES

An error message is displayed when an error is detected in the PTA command or program.

There are two types of errors;an execution-stop error (fatal: F) and an execution-continuable error (warning: W).

- Execution-stop error (F:Fatal) :
  This type of error stops the execution of the program unconditionally.
- Execution-continuable error (W:Warning error) :
  When there is no ERROR statement in the line next to the line where this type of error occurs, the execution stops; but if there is an ERROR statement, execution continues.
  And also, error interruption process can continue the execution.

Error Message Format

The error message is displayed in the following format.

- PTA program:

```
ERROR Error level Error number[,Error-occurrence line number]
```

This is displayed at the program execution.

- PTA library:

```
ERROR Error level Error No.[,erred line No.,erred program name]
```

This is displayed at program execution.

No.300 and on are errors of the library program itself.

ERROR Statement

(1) Function

For an execution-continuable error generated at program execution, execution can be continued by using the ERROR statement.

An ERROR statement can be programed over several lines.

(2) Format

```
ERROR(21Ø,1ØØØ)
```

21Ø: Error number

1ØØØ: Next program line to be executed

This statement means that when the error (generated in the previous line) corresponds to the error number 210,the program of line 1000 is executed.

When it does not correspond, the error message is displayed and execution stops.

(3) Example

```
1Ø X = Ø
2Ø Y = 1ØØ/X
3Ø ERROR(21Ø,1ØØ)   ;  If the  error (210: the divisor is 0) occurs, jump to line 100.
4Ø Y = Y+5Ø
  ⋮
```

ERRMAIN Statement

(1) Function

To branch to the main routine whenever a execution continuable ERROR occurred, use the ERRMAIN statement.

(2) Format

```
ERRMAIN (error number)
```

(3) Example

```
1Ø   INPUT A
2Ø   GOSUB 1ØØØ
3Ø   :
     :
1ØØØ WRITE 217,A
1Ø1Ø ERRMAIN(222)      ; If the error (222) occurs because the data of WRITE statement can
1Ø2Ø                     not output, the program returns to the main routine.
     :
```

Note: If the ERRMAIN statement has been executed in the highest level of the routine, ERROR 213 is generated.

Error Processing Subroutines

ON ERROR statement

(1) Function

Registers the subroutine to branch (interrupt) to when an error occurs.

(2) Format

```
ON ERROR line number(or *label)
```

After executing this statement and an error that is possible to continue execution occurs, an interrupt occurs and the error processing subroutine is executed from the line number (or label) specified.

OFF ERROR statement

(1) Function

Releases the registered subroutine to branch (interrupt) to when an error occurs.

(2) Format

```
OFF ERROR
```

After executing this statement, error interrupts will not occur.

Returning from error processing subroutines (RETERR, RETRY, RESUME and GIVE UP statements)

(1) Function

Returns from an error interrupt.

(2) Format

`RETERR`	(Continues from the statement following the statement where the error occurred.)
`RETRY`	(Continues reexecuting from the statement caused the error.)
`RESUME`	(Continues from specified line.)
`GIVEUP`	(Stops program execution.)

Note: See Section 4, "RETERR statement" to "GIVEUP statement".

ERRREAD (m) function

(1) Function

Reads the line where the error occurred and the error code in the middle of an error processing subroutine.

(2) Format

`V=ERRREAD(Ø)`	(Error code)
`V=ERRREAD(1)`	(Line where error occurred)

(3) Example

`1ØØ ON ERROR 2ØØ`	; Jumps to line 200 on error
`11Ø INPUT X`	
`12Ø Y=1ØØ/X`	
`13Ø PRINT Y`	
`14Ø GOTO 11Ø`	
`15Ø STOP`	
`2ØØ C=ERRREAD(Ø)`	
`21Ø IF C=21Ø GOSUB 3ØØ`	; For "Divide by zero", continues to execute from line 130 displaying "ERROR/0".
`22Ø IF C<>21Ø GIVEUP`	; On other errors, stops program execution
`23Ø RETERR`	
`3ØØ PRINT "ERROR /Ø"`	
`31Ø RETURN`	

Error List

Table 8-1 shows the error number and error cause. In the table, F (Fatal) denotes the execution-stop error and W (Warning) denotes the execution-continuable error.

Table 8-1 PTA Error List

Error No.	Cause of error	W,* F**
0	[↵] key pressed but no commands or statement input	F
1	Number of characters (representing variable) exceeds 8, or number of characters (representing program name) exceeds 6.	W
2	Format of numeric constant in correct Example : `Ø..1 4.5EE2`	W
3	Too many input digits, or value of numeric constant too large or too small (Format of numeric constant incorrect)	W
4	Format of character string constant incorrect Example : `A$="ABC`	W
5	Format incorrect Example : `PRINT A:G6.2`	W
6	Statement cannot be interpreted (command format error) Example : `GOTO ABC`	W
7	Statement insufficiently described Example : `GOTO`	W
8	Statement excessively described Example : `GOTO 1ØØ,2ØØ`	W
9	Number of variables exceeds 256 (Up to 256 user-defined variables can be written)	W
10	Character cannot be interpreted Example : `-1ØØ`	W
11	Format (of binary or hexadecimal constant) incorrect Example : `8#=# 11Ø`	W
12	Value (of binary or hexadecimal constant) too large Binary constant : up to 8 characters Hexadecimal constant : up to 2 characters Example : `8#=#1ØØØØØØØØ`	W
13	Number of format digits too large Example : `PRINT A:F6.5`	W

*W : Execution-continuable error (Warning)
**F : Executiong-stop error (Fatal error)

Table 8-1 PTA Error List (Continued)

Error No.	Cause of error	W, F
14	Command operand cannot be interpreted Example : `LIST A,B`	W
15	Command operand insufficient Example : `LISTG`	W
16	Command operand excessive Example : `DELETE 10,100,300`	W
17	Line number exceeds 65535 (Program line number is 1 to 65535)	W
20	Program on a line too long to assemble	W
21	Undefined-line-number label used as command operand	W

Note : Errors 0 to 21 may occur during program input or command execution.
Errors 6 to 8, however, may also occur during statement execution.

Table 8-1 PTA Error List (Continued)

Error No.	Cause of error	W, F
101	Value of command operands 1 and 2 incorrect Example : `LIST 1ØØ,1Ø`	F
102	Program exceeds memory capacity	F
103	No Line number or program, designated by command (LIST, LISTG, DELETE, RENUM, and SAVE commands)	F
104	Since number of GOTO or GOSUB statements excessive (>100), RENUM statement cannot be executed	F
105	Since line number (specified by GOTO or GOSUB operand) not found, RENUM statement cannot be executed	F
111	Line number exceeds 65535 when RENUM and PCOPY statements executed	F

Note : Errors 101 to 105 and 111 may occur during command execution.

Table 8-1 PTA Error List (Continued)

Error No.	Cause of error	W, F
120	Media write-protected	W
121	Media not installed	W
122	Media memory overflow	W
123	Specified program not stored in media	W
124	Media faulty	W
125	Memory type incorrect	W
126	Media formatting incorrect	W
127	Media not formatted	W
150	Label is not defined or defined more than once	F
151	No DATA statement	F
180	Error of the command transmitted from PTA to main frame	W

Note : Errors 120 to 127 may occur when a command or statement attempts to access the media (PMC or FD).

Table 8-1 PTA Error List (Continued)

Error No.	Cause of error	W, F
201	Program cannot be resumed (CONT command)	F
202	Specified line number missing RUN command executed without program (RUN, CONT commands and GOTO, GOSUB statements)	W
203	Array subscript (in DIM statement) incorrect (The array subscript must be from 1 to 1024; the bit array subscript must be from 1 to 8, and the character array subscript must be from 1 to 255.)	W
204	Used as simple, or system variables before array declaration by DIM statement	W
205	Array declaration overlapped	W
206	Insufficient variable memory capacity due to program memory overflow	F
207	Arithmetic operation of character data or bit data	W
208	Data-type combination incorrect for conversion	W
209	Overflow or underflow occurred	W
210	Divide by 0	W
211	Value of arithmetic function parameter too large or too small	W
212	Nesting (by subroutine, FOR and NEXT statement) exceeded 10 levels	F
213	No return destination specified for RETURN statement	F
214	Comparison cannot be made by IF statement Right and left side data-type combination incorrect	W

Table 8-1 PTA Error List (Continued)

Error No.	Cause of error	W, F
215	SOS statement is executed	F
216	No corresponding FOR statement. That is, there are excess NEXT statements. (RUN, CONT command and GOTO, GOSUB statements)	W
217	Input data format (in INPUT statement) incorrect	W
218	Input data (in INPUT statement) insufficient	W
219	Excess amount or too large input data in INPUT statement	W
220	Minus sign used in exponentiation Example : `-1!5`	W
221	Data can not be input in GPIB (Talker device not connected)	W
222	Data cannot be output in GPIB	W
223	Parameter (in the statement) outside range or variable type incorrect Example : `WAIT A$`	W
224	Simple variable includes array subscript	W
225	Array variable has no subscript	W
226	Array-variable subscript out of boundary Note that the subscript range declared in DIM J(5) is J(0) to (4).	W
227	GPIB execution is impossible because the PTA is set as the device	W
228	GPIB execution is impossible because the PTA is set as the controller	W

Table 8-1 PTA Error List (Continued)

Error No.	Cause of error	W, F
229	STOP statement (to terminate program execution) not specified	W
230	Attempt made to refer to non-referable system variable	W
231	Attempt made to assign non-assignable system variable	W
232	Array variable subscript not numeric	F
233	Parameter (in boolean function) not bit type	W
234	Parameter of FOR statement is character or bit type	W
235	The I/O type specification in the EVENT statement is out of range (0 to 99).	W
236	Variable of NEXT statement does not correspond to that of FOR statement specified before NEXT statement Example : `3Ø FOR C=...` `9Ø NEXT D`	W
237	Six or more character constants and variables used in INPUT, PRINT, READ or WRITE statement Example : `PRINT"FREQ",F(C),"Hz","LEVEL",LEV,"dBm"`	W
238	Variable type and format type of PRINT or WRITE statement do not agree	W
239	Operand (in LISTG, WRITE or READ statement) outside range (0 to 31) Example : `LISTG 35`	W
240	Variable or constant values of CALL statement or system function outside range	W
241	Vairable or constant type of CALL statement or system function incorrect	W
242	System variable used in CALL statement or system function	W

Table 8-1 PTA Error List (Continued)

Error No.	Cause of error	W, F
243	The RETURN or RETMAIN statement was used to return from event or error interrupt processing.	F
244	Media data file not open	W
245	Media data file opened	W
246	Media data already read	W
247	Media data type and variable type combination incorrect (unconvertible)	W
248	Excess amount or too large input data value in READ statement.	W
249	Insufficient input data in READ statement	W
250	Input data format (in READ statement) incorrect	W
251	The RETINT statement was used for something other than event interrupt processing. Or, the GOSUB statement was executed in the middle of event interrupt processing and the RETURN statement to return was not executed, but the RETINT statement was instead.	F
252	The RETERR, RETRY, RESUME, GIVEUP statements were used for something other than error interrupt processing. Or, the GOSUB statement was executed in the middle of error interrupt processing. Or, the RETURN statement to return was not used and one of the above statements was executed.	F
253	The ERRREAD function was executed for something other than error interrupt processing.	F
254	The STATUS function was executed for something other than event interrupt processing.	F

Table 8-1 PTA Error List (Continued)

Error No.	Cause of error	W, F
301	Library/program is being selected.	W
302	The specified measuring instrument library does not exist in the memory.	W
303	A program having the new program name specified by RENAME exists.	F
304	The file containing the same name as that of the program in execution was loaded.	W
305	The number of nesting by CALLIB has exceeded 10.	F
306	The library was executed during sequence registering/downloading.	F
307	The specified measuring instrument library is being executed.	W
308	The specified measuring instrument library is being locked.	W
309	Result of processing by the main frame’s measuring instrument is abnormal.	W
310	The library is being registered.	W
311	The LIBRARY statement cannot be edited.	W
312	CHKFILE was executed to the .MNU file.	W
313	The specified measuring instrument library resides in ROM.	W
314	The COMCLEAR statement cannot be executed in the nested PTA library.	W