

PART 2 PROGRAMMING MANUAL

(GPIB OPERATION)

(Rev.12)

How to Use This Manual

This manual Part 2 describes the controller handling procedures, plus the GPIB remote control operating procedures and BASIC programming for users who already have a certain amount of knowledge and experience in programming with the BASIC language.

Network Analyzer can be remote-controlled by any of the following three methods.

- ① Remote control by external controller (Refer to Chapter 2.)
- ② Activation of built-in BASIC programming functions, and exchange data with an external controller while controlling the network analyzer. (Refer to Chapter 3, 4, and 5.)
- ③ Activation of built-in BASIC programming functions, and controlling external devices and the network analyzer itself as the GPIB controller. (Refer to Chapter 3, 4, and 5.)

Notes

1. Part 2 explains the following products (modes).

Applicable models : R3751AH/BH/EH
R3761A/B/E
R3762A/AH/B/BH/E
R3763A/B

2. In this manual, the mode names are abbreviated as follows.

R3751AH/BH/EH -----	R3751	} network analyzer
R3761A/B/E -----	R3761	
R3762A/AH/B/BH/E --	R3762	
R3763A/B -----	R3763	

NETWORK ANALYZER
PROGRAMMING MANUAL

Table of Contents

TABLE OF CONTENTS

1. INTRODUCTION	1 - 1
1.1 Outline	1 - 1
1.2 GPIB Modes	1 - 2
2. REMOTE CONTROL BY GPIB EXTERNAL CONTROLLER	2 - 1
2.1 Outline	2 - 1
2.2 GPIB Functions	2 - 2
2.3 GPIB Addressing	2 - 3
2.4 GPIB Input and Output Formats	2 - 5
2.4.1 Outline	2 - 5
2.4.2 Permissible Input Characters	2 - 6
2.4.3 Address Specification	2 - 7
2.4.4 Output Format	2 - 7
2.4.5 Input Formats	2 - 8
2.4.6 GPIB Code Table	2 - 9
2.5 Service Request	2 -31
2.6 Program Examples	2 -32
2.6.1 Program for Determining Difference between Very Large and Very Small Points within Same Specified Frequencies, and Maximum Value of Difference Between Adjacent Inflection Points	2 - 32
2.6.2 Trace Data Input/Output	2 - 34
2.6.3 Limit Line Output	2 - 36
2.6.4 SRQ	2 - 37
2.6.5 Starting BASIC from External Controller	2 - 39
2.6.6 Data Sending/Receiving with SRQ	2 - 42
2.6.7 Program Example Using External Controller or Built-in Basic .	2 - 45
2.6.8 X'TAL Filter Measuring Program Example	2 - 47
2.6.9 Example of Measuring Program Using Parallel I/O Ports	2 - 50
2.6.10 Example of Program Where Limited Test Function Is Used in Low-pass Filter Measurements	2 - 54
2.6.11 AUTO SCALE	2 - 61
2.6.12 Binary Data Input and Output	2 - 62
3. CONTROL MODE	3 - 1
3.1 Outline	3 - 1
3.2 Setting Controller Mode	3 - 2
3.3 Handling Floppy Disks	3 - 3
3.4 File Management	3 - 7
3.4.1 Outline	3 - 7
3.4.2 Saving and Recalling Programs	3 - 7
3.4.3 Initialization of Floppy Disk	3 - 8
3.4.4 File Management	3 - 8
3.4.5 File Storage	3 - 9
3.4.6 File Recalling	3 - 9
3.4.7 File Deletion	3 - 9
3.4.8 File Name Change	3 - 9

NETWORK ANALYZER
PROGRAMMING MANUAL

Table of Contents

4. BASIC PROGRAMMING	4 - 1
4.1 Outline	4 - 1
4.2 Activation of Program Mode	4 - 2
4.3 Editor Mode Activation	4 - 5
4.4 Program Editor Keys	4 - 6
4.5 Program Editing	4 - 11
4.6 Programming Rules	4 - 14
4.6.1 Program Architecture	4 - 14
4.6.2 Objects	4 - 16
4.6.3 Operators	4 - 23
5. COMMAND AND STATEMENT SYNTAX AND COMMENTARY	5 - 1
5.1 Outline	5 - 1
5.2 List of Commands and Statements	5 - 2
5.3 BASIC Command Syntax	5 - 5
5.4 BASIC Statemet Syntax	5 - 38
5.5 BASIC GPIB Control Statement Syntax and Activity	5 - 85
5.6 Syntax of BASIC File Control Statement	5 -101
6. BUILT-IN FUNCTIONS	6 - 1
6.1 Outline	6 - 1
6.2 List of Built-in Functions	6 - 3
6.3 Description of Built-in Function	6 - 7
6.3.1 Function Determining Data on the Horizontal Axis	6 - 8
6.3.2 Function Determining Response Value	6 - 11
6.3.3 Functions Which Include Search Functions	6 - 13
6.3.4 Band Width Calculation Function	6 - 16
6.3.5 Ripple Function	6 - 20
6.3.6 Other Functions	6 - 36
APPENDIX	
A1.1 Error Message	A1 - 1
A1.1 How to Display an Error Message	A1 - 1
A1.2 How to Display the Present Position of Program	A1 - 1
A1.3 Error Messages	A1 - 1
INDEX	I - 1

NETWORK ANALYZER
PROGRAMMING MANUAL

List of Illustrations

LIST OF ILLUSTRATIONS

<u>No.</u>	<u>Title</u>	<u>Page</u>
2 - 1	Status Register	2 - 31
3 - 1	Floppy Disk Dimensions and Component Parts	3 - 3
3 - 2	Floppy Disk Insertion Method (R3751)	3 - 4
3 - 3	Floppy Disk Write Protect and Write Enable	3 - 6
4 - 1	CRT Display During Program Mode	4 - 2
4 - 2	CRT Display During Editor Mode	4 - 5
6 - 1	Details of Response Type	6 - 2

NETWORK ANALYZER
PROGRAMMING MANUAL

List of Tables

LIST OF TABLES

<u>No.</u>	<u>Title</u>	<u>Page</u>
2 - 1	How to Read the GPIB Code Table (1 of 2)	2 - 9
2 - 2	GPIB Program Code	2 - 11
4 - 1	CTRL Key Operation	4 - 6
4 - 2	Function Operations	4 - 8
4 - 3	List of Key Words	4 - 15
4 - 4	Alphanumeric Characters	4 - 18
6 - 1	Type of Response to Built-in Function	6 - 1

NETWORK ANALYZER
PROGRAMMING MANUAL

1.1 Outline

1. INTRODUCTION

1.1 OUTLINE

The purpose of this manual Part 2 is to describe the procedures for controlling Network Analyzer and external peripherals using the analyzer's GPIB remote control and built-in BASIC controller functions.

Network Analyzer includes the IEEE standards 488-1978 metering bus GPIB (General Purpose Interface Bus) as a standard feature to enable remote control by external controller. And using the controller functions and functions included in the built-in BASIC language, device characteristics can be tested at high speed, and smallscale GPIB systems can be readily constructed.

NETWORK ANALYZER
PROGRAMMING MANUAL

1.2 GPIB Modes

1.2 GPIB Modes

Network Analyzer operates in the following two modes.

① TALKER/LISTENER Mode

TALKER/LISTENER is the normal mode controlled by external controller. Data can be exchanged with the external controller while running a built-in BASIC program.

② System Controller Mode

System controller mode enables Network Analyzer measuring functions and external equipment connected to Network Analyzer to be controlled by builtin BASIC program.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.1 Outline

2. REMOTE CONTROL BY GPIB EXTERNAL CONTROLLER

2.1 Outline

GPIB is an interface system designed to connect measuring equipment to the controller and peripheral devices by simple cable connections. In comparison to more conventional interface systems, GPIB features greater expandability, plus electrical, mechanical, and functional compatibility with other equipments and other brands.

The GPIB system includes three roles - controller, TALKER, and LISTENER, and when controlled by an external GPIB controller, Network Analyzer retains the TALKER and LISTENER functions.

CAUTION

When a BASIC program is run in TALKER/LISTENER mode, settings cannot be made by GPIB command from the external controller (due to priority given to BASIC ENTER and OUTPUT commands).

To make settings by GPIB command from external controller, the BASIC program must first be stopped.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.2 GPIB Functions

2.2 GPIB Functions

- SH1: Source handshake function
- AH1: Accept handshake function
- T6: Basic TALKER function, serial polling function, and TALKER function cancellation by LISTENER designation
- TE0: No expanded TALKER function
- L4: Basic LISTENER function, and LISTENER function cancellation by TALKER function designation
- LE0: No expanded LISTENER function
- SR1: Service request function
- RL1: Remote function, local function, local lockout function
- PP0: No parallel polling function
- DC1: Device clear function
- DT1: Device trigger (when in hold mode)
- C0: No controller function (when in TALKER/LISTENER mode)
- C1: System controller function (when in controller mode)
- E1: Use open collector bus driver

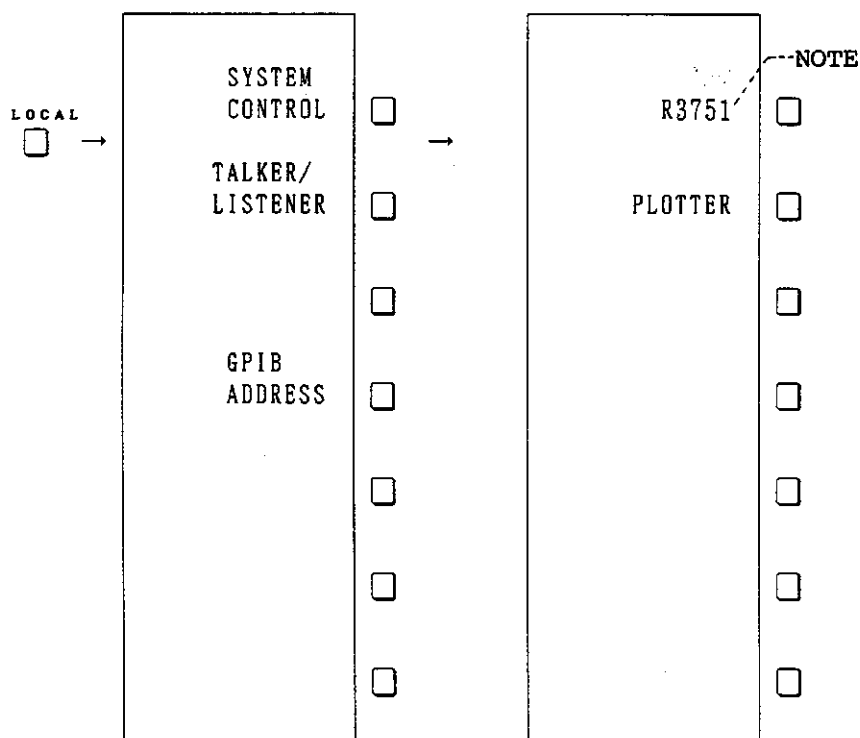
NETWORK ANALYZER
PROGRAMMING MANUAL

2.3 GPIB Addressing

2.3 GPIB Addressing

[Gpib Handling Procedures]

When the LOCAL key is pressed and the GPIB address is selected, the softkey menu changes as displayed below.



Note : Display depends on the model for use.
R3751 is displayed if R3751AH/BH/EH is used.
R3761 is displayed if R3761A/B/E is used.
R3762 is displayed if R3762A/B/E is used.
R3763 is displayed if R3763A is used.

- GPIB address is set when the Network Analyzer key is pressed.

GPIB address can be set to any value from 0 to 30.
Following input of a number by the relevant numeric key, the GPIB address is set by pressing the deg keys for R3751, and the ENT key for R3761, R3762 and R3763.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.3 GPIB Addressing

- The plotter GPIB address is set by pressing the PLOTTER key.

Plotter address can be set to any value from 0 to 30.

Following input of a number by the relevant numeric key, the plotter address is set by pressing the deg keys for R3751, and the ENT key for R3761, R3762 and R3763.

(This address is valid only in system controller mode.)

CAUTION

- Do not specify the same address as the GPIB address for an external controller and other connected devices.
- The address specified here is the address for controlling the network analyzer by using an external controller. The address for controlling the network analyzer by built-in BASIC program is fixed at "31".

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

2.4 GPIB Input and Output Formats

2.4.1 Outline

(1) Address specification

The network analyzer serves as a GPIB controller controlling the network analyzer itself and external equipment. Address specification is as follows.

0 to 30 : To control external equipment
31 : To control the network analyzer itself
33 to 37: To control the parallel I/O instruments

(2) Input type

Basically, the GPIB code can be input in the same way as panel operation.

When 1 MHz center frequency is set, the following operation is required.

● Panel operation

- | |
|-------------------------|
| ① Press the CENTER key. |
| ② Press numeric key 1. |
| ③ Press unit key MHz. |

● GPIB code input

OUTPUT 31;" CENTERF 1 MHz"
① ↑ ② ↑ ③ ↑
Code Data Unit

As described above, the network analyzer can be controlled. When external equipment is controlled, change the address and input the GPIB code of external equipment. For details, see Item 2.3 and 2.4.

Note : The GPIB code consists of capital letters only.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

(3) Response type

This type is used to check the value indicated by marker. A program example is as follows.

```
100 OUTPUT 31;"MKR1A?"  
110 ENTER 31;F,L  
120 PRINT F,"HZ",L,"dB"
```

Like line 100, only "?" is added to the GPIB code. This is a request for data output. It is input with the ENTER statement for line 110. It can be printed like line 120.

Note that the details of GPIB code is different. Especially, a response of marker value depends on measuring format. For details, see 2.4.6 GPIB Code Table or *29, *30, and *31.

Note : When several pieces of data is returned, unnecessary data is also received.

2.4.2 Permissible Input Characters

Although ASCII characters are recognized by the network analyzer all those apart from the characters listed below are disregarded in normal operations except label input mode.

- ① Upper case alphabetic characters
- ② Numeric characters
- ③ Decimal point
- ④ + or -
- ⑤ , (comma)
- ⑥ ; (semi colon)
- ⑦ CR (carriage return) : Recognized only as GPIB delimiter
- ⑧ LF (line feed) : Recognized only as GPIB delimiter

Note 1: All leading zeros are disregarded. 000208640 → 208640

Note 2: All lower case characters are disregarded.

STARTFrequency|MHz → STARTF|MHz

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Note 3: Numeric character inputs may include decimal point and exponential expressions.

0 thru 9 Mantissa may include sign and up to 17 significant digits.
. + -
E Exponential part may include sign and one or two significant digits.

2.4.3 Address Specification

In addition to the BASIC program incorporated in the network analyzer and the external GPIB controller, the following codes are used to control the network analyzer.

When the network analyzer is controlled in the TALKER/LISTENER mode using the built-in BASIC program, use instrument address 31.

OUTPUT 31 ;
ENTER 31 ;

When the parallel I/O port is input or output, instrument address 33 to 37 for OUTPUT and 34 to 37 for ENTER are used.

OUTPUT 33 ; ENTER 34 ;
OUTPUT 34 ; ENTER 35 ;
OUTPUT 35 ; ENTER 36 ;
OUTPUT 36 ; ENTER 37 ;
OUTPUT 37 ;

2.4.4 Output Format

- ① Numeric Values (integers) in ASCII Code
- ② Floating Decimal Point Numeric Values in ASCII Code

±D. DDDDDDDDDDDDDDE±DD
Total number of characters 22
Mantissa sign - (minus) + (plus)
One digit (mantissa and number of digits to left of decimal point)
 + decimal point + 15 digits (mantissa and number of
 digits to right of decimal point)
E Exponent
Exponential part sign - (minus) + (plus)
Two digits exponential part

Example : 1.123456789012345E+08

CAUTION

Although there is no unit code output, and internal basic unit is used. Hz, V, dB, m, Sec, Unit, div, %, deg, etc.

2.4.5 Input Formats

(1) General Format

[Code] [Additional code] [Data] [Unit] [Terminator]

① [Code]

Basic mnemonics for the network analyzer

② [Additional Code]

Designation used for switches qualifying basic mnemonic or to indicate one of several types.

- ON/OFF
- Integer value which selects one of several types

③ [Data]

Data set in function specified by code

- Numeric value (ASCII)
 - Integer : 278 etc.
 - Real number : 278.0, -256.8E+2 etc.
 - Character string (ASCII)
 - String enclosed between double quotation marks: "278" etc.

④ [Unit]

All data must have a unit.

⑤ [Terminator]

Any of the following four types can be specified.

(CR) (LF) + EOI

(LF)

Final byte + EOI

(CR) (LF) Initial status type

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

(2) Input Format Types

- ① TYPE1 : [Code] [Terminator]
- ② TYPE2 : [Code] [Additional code] [Terminator]
- ③ TYPE3 : [Code] [Data] [Unit] [Terminator]
- ④ Enquiry type : [Code] [?]

2.4.6 GPIB Code Table

The method for reading the GPIB Code Table is outlined in Table 2-1.

Table 2 - 1 How to Read the GPIB Code Table

Item	Function
Code	Program setting code
Contents	Code function
Description format	Input format [t] : [Code] [Terminator] [s] [t] : [Code] [Additional code] [Terminator] [d] [u] [t] : [Code] [Data] [Unit] [Terminator] Additional code ON or OFF (ASCII) Numeric value (ASCII) (ASCII) Data (ASCII) Terminator GPIB terminator (CR, LF)

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 1 How to Read the GPIB Code Table (cont'd)

Item	Function
Response format	<p>Response to enquiry about setting condition</p> <p>1, 0 : ON/OFF or YES/NO</p> <p>D : Data</p> <p style="padding-left: 2em;">D Numeric value</p> <p style="padding-left: 2em;">Contnets of []</p> <p style="padding-left: 2em;">s Data on horizontal axis</p> <p style="padding-left: 4em;">: FORMAT valid in all modes</p> <p style="padding-left: 2em;">r Data on vertical axis</p> <p style="padding-left: 4em;">: FORMAT valid in all modes</p> <p style="padding-left: 2em;">i Data on vertical axis (AUX)</p> <p style="padding-left: 4em;">: FORMAT valid only when Smith or Polar or parameter conversion is ON</p> <p style="padding-left: 2em;">lc ... L[H] or C[F]</p> <p style="padding-left: 4em;">: FORMAT valid only when Smith or parameter conversion is ON</p> <p style="padding-left: 2em;">C Operation data</p> <p>The i and lc values are not returned when FORMAT setting is not valid.</p> <p>When partitioned by a comma (,) such as D(s,r), the output is also partitioned by comma in the GPIB.</p>
Remarks	<p>In Notes 1 to 4</p> <p>Note that the GPIB code functions depending on the type of the network analyzer.</p> <p>*1 to *34</p> <p>Note the GPIB code.</p>

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code

Item	Code	Contents	Descriptive format	Response format	Remarks
● ACTIVE CHANNEL See chapter 3 of this manual Part 1 for description of basic functions.					
CHANNEL	CH1	CH1 active	[t]	1, 0	
	CH2	CH2 active	[t]	1, 0	
● INPUT MEASURE See chapter 3 of this manual Part 1 for description of basic functions.					
INPUT PORTS	ARIN	A/R	[t]	1, 0	Note 2
	BRIN	B/R	[t]	1, 0	Note 1
	ABIN	A/B	[t]	1, 0	Note 1
	AIN	A	[t]	1, 0	Note 4
	BIN	B	[t]	1, 0	Note 4
	RIN	R	[t]	1, 0	Note 4
	S11	REFLECTION	[t]	1, 0	Note 5
	S21	TRANSMISSION	[t]	1, 0	Note 5
PARAMETER CONVERSION	CONVRZ	Z(Reflection)	[t]	1, 0	
	CONVRY	Y(Reflection)	[t]	1, 0	
	CONVOFF	OFF	[t]	1, 0	
	SETZO	Z0	[d][u][t]	D	
S-PARAMETER	S11	TEST SET control	[t]	1, 0	Note 1
	S12	TEST SET control	[t]	1, 0	Note 1
	S21	TEST SET control	[t]	1, 0	Note 1
	S22	TEST SET control	[t]	1, 0	Note 1
● FORMAT See chapter 3 of this manual Part 1 for description of basic functions.					
FORMAT	LOGMAG	Log Mag	[t]	1, 0	
	PHASE	Phase	[t]	1, 0	
	DELAY	Delay	[t]	1, 0	
	SRJX	Smith (R+jX)	[t]	1, 0	
	SGJB	Smith (G+jB)	[t]	1, 0	
	POLAR	Polar	[t]	1, 0	
	LINMAG	Lin Mag	[t]	1, 0	
	SWR	VSWR	[t]	1, 0	Note 3
	REAL	Real	[t]	1, 0	
	IMAG	Imag	[t]	1, 0	
	UNWRAP	Phase (-∞, +∞)	[t]	1, 0	

- Note 1 : Enable for type A only
- Note 2 : Enable for type A and B
- Note 3 : Enable for R3761/3762
- Note 4 : Enable for R3751 only
- Note 5 : Enable for R3763 only

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

Item	Code	Contents	Descriptive format	Response format	Remarks
● GROUP DELAY APERTURE		See chapter 3 of this manual Part 1 for description of basic functions.			
APERTURE	APERTP	Data Entry	{d}{u}{t}	D(%)	
● SCALE REF.		See chapter 3 of this manual Part 1 for description of basic functions.			
SCALE	AUTO	Auto Scale	{t}	
	SDIV	/Division	{d}{u}{t}	D(r)	
REFERENCE	REFV	Ref. Value	{d}{u}{t}	D(r)	
	REPP	Ref. Position	{d}{u}{t}	D(%)	
	REPL	Ref. Line on/off	{s}{t}	1, 0	
UP SCALE	UPSCAL	on/off	{s}{t}	1, 0	
● DISPLAY		See chapter 3 of this manual Part 1 for description of basic functions.			
CHANNEL	DUAL	Dual on/off	{s}{t}	1, 0	
	SPLIT	Split on/off	{s}{t}	1, 0	
GRATICULE	GRAT	Graticule on/off	{s}{t}	1, 0	
CRT	INTENS	Intensity	{d}{u}{t}	D	
DISPLAY	DISPDATA	Data	{t}	1, 0	
	DISPDM	Data & Memory	{t}	1, 0	
	DTOM	Data to Memory	{t}	1, 0	*2
DATA/MEM	DISPDDM	on/off	{s}{t}	1, 0	*6
LABEL	LABEL	LABEL	{strings}{t}	...	*8

*2 : Response of 1 if MEM already stored, but 0 if not.

*6 : ON not possible if MEM not stored.

*8 : Append character string after GPIB code.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

Item	Code	Contents	Descriptive format	Response format	Remarks
● SOURCE See chapter 3 of this manual Part 1 for description of basic functions.					
FREQUENCY	STARTF	Start freq.	[d][u][t]	D(s)	
	STOPF	Stop freq.	[d][u][t]	D(s)	
	CENTERF	Center freq.	[d][u][t]	D(s)	
	SPANF	Span freq.	[d][u][t]	D(s)	
OUTPUT PORTS	PORT1	Output port1	[t]	1, 0	Note 4
	PORT2	Output port2	[t]	1, 0	Note 4
OUTPUT LEVEL	OUTLEV	Output level	[d][u][t]	D(r)	
FREQ. STEP	FSTPA	Freq. step auto	[t]	1, 0	
	FSTPM	Freq. step manual	[t]	1, 0	
STEP SIZE	FRQSTP	Freq. step	[d][u][t]	D(s)	*11
S PARAMETER TEST SET ATTENUATOR	ATTP1	PORT1 ATT	[d][u][t]	D	Note 1
	ATTP2	PORT2 ATT	[d][u][t]	D	Note 1
● SWEEP See chapter 3 of this manual Part 1 for description of basic functions.					
TIME	STIME	Sweep time	[d][u][t]	D(t)	
TYPE	COUPLE	Couple on/off	[s][t]	1, 0	
	LINFREQ	Lin freq.	[t]	1, 0	
	LOGFREQ	Log freq.	[t]	1, 0	
	CW	CW	[t]	1, 0	
	LEVEL	Level sweep	[t]	1, 0	
	PARTIAL	PARTIAL on/off	[s][t]	1, 0	
	USRSWP	User sweep	[t]	1, 0	
POINTS	M1201P	1201 Points	[t]	1, 0	
	M601P	601 Points	[t]	1, 0	
	M301P	301 Points	[t]	1, 0	
	M201P	201 Points	[t]	1, 0	
	M101P	101 Points	[t]	1, 0	
	M51P	51 Points	[t]	1, 0	
	M21P	21 Points	[t]	1, 0	
	M11P	11 Points	[t]	1, 0	
	M6P	6 Points	[t]	1, 0	
	M3P	3 Points	[t]	1, 0	

Note 1 : Enable for type A only

Note 4 : Enable for R3751 only

*11 : When setting FSTPA, a value 1/10th of SPAN is automatically set instead.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

Item	Code	Contents	Descriptive format	Response format	Remarks
TRIGGER	FREE	Internal	{t}	1, 0	
	LINE	Line	{t}	1, 0	
	EXTERN	External	{t}	1, 0	
MODE	CONT	Continue	{t}	1, 0	
	SINGLE	Single	{t}	1, 0	
	SWPHLD	Sweep HOLD	{t}	1, 0	
RESTART	MEAS	Restart	{t}	---	*3
● PARTIAL SWEEP DATA ENTRY		See chapter 3 of this manual Part 1 for description of basic functions.			*4
PARTIAL SWEEP DATA ENTRY	PSEGCL	Segment clear	{t}	1, 0	*5
	PSEG	Segment No.	{d}{u}{t}	D	
	PSTART	Start freq.	{d}{u}{t}	D(s)	
	PSTOP	Stop freq.	{d}{u}{t}	D(s)	
● USER SWEEP DATA ENTRY		See chapter 3 of this manual Part 1 for description of basic functions.			*4
USER SWEEP DATA ENTRY	USEGCL	Segment clear	{t}	1, 0	*5
	USEG	Segment No.	{d}{u}{t}	D	
	USTART	Start freq.	{d}{u}{t}	D(s)	
	USTOP	Stop freq.	{d}{u}{t}	D(s)	
	UFREQ	freq.	{d}{u}{t}	D(s)	
	UPOINT	Points	{d}{u}{t}	D	
● LEVEL SWEEP DATA ENTRY		See chapter 3 of this manual Part 1 for description of basic functions.			
LEVEL SWEEP DATA ENTRY	STLEVEL	Start level	{d}{u}{t}	D(r)	
	SPLEVEL	Stop level	{d}{u}{t}	D(r)	
CW FREQUENCY	CWFREQ	CW Frequency	{d}{u}{t}	D(s)	

*3 : Sweep from beginning.

*4 : Partial sweep ON/OFF is selected in type column.

*5 : Response of latest setting

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

Item	Code	Contents	Descriptive format	Response format	Remarks	
● CW SWEEP DATA ENTRY						
CW freq.	CWF		[d][u][t]	D(s)		
● RECEIVER See chapter 3 this manual Part 1 for description of basic functions.						
IMPEDANCE ATTENUATOR	RI50A20	R50Ω, 20dB	[t]	1. 0	Note 4	
	RI50A0	R50Ω, 0dB	[t]	1. 0	Note 4	
	RI1A20	R1MΩ, 20dB	[t]	1. 0	Note 4	
	RI1A0	R1MΩ, 0dB	[t]	1. 0	Note 4	
	AI50A20	A50Ω, 20dB	[t]	1. 0	Note 4	
	AI50A0	A50Ω, 0dB	[t]	1. 0	Note 4	
	AI1A20	A1MΩ, 20dB	[t]	1. 0	Note 4	
	AI1A0	A1MΩ, 0dB	[t]	1. 0	Note 4	
	BI50A20	B50Ω, 20dB	[t]	1. 0	Note 4	
	BI50A0	B50Ω, 0dB	[t]	1. 0	Note 4	
	BI1A20	B1MΩ, 20dB	[t]	1. 0	Note 4	
	BI1A0	B1MΩ, 0dB	[t]	1. 0	Note 4	
	RBW	RBW1KHZ	1KHz	[t]	1. 0	
		RBW300HZ	300Hz	[t]	1. 0	
		RBW100HZ	100Hz	[t]	1. 0	
		RBW30HZ	30Hz	[t]	1. 0	
RBW10HZ		10Hz	[t]	1. 0		
CLEAR TRIP	CLRTRIP	Clear trip	[t]	---	Note 4	
● AVERAGE See chapter 3 of this manual Part 1 for description of basic functions.						
AVERAGING	AVERAGE	off	[s][t]	1. 0		
	AVR2	2	[t]	1. 0		
	AVR4	4	[t]	1. 0		
	AVR8	8	[t]	1. 0		
	AVR16	16	[t]	1. 0		
	AVR32	32	[t]	1. 0		
	AVR64	64	[t]	1. 0		
	AVR128	128	[t]	1. 0		

Note 4 : Enable for R3751 only

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

Item	Code	Contents	Descriptive format	Response format	Remarks
● CALIBRATION		See chapter 3 of this manual Part 1 for description of basic functions.			
NORMALIZE (THRU)	NORM	Normalize(Thru) ON/OFF	[s][t]	1, 0	
NORMALIZE (SHORT)	NORMS	Normalize(Short) ON/OFF	[s][t]	1, 0	
CALIBRATION	CORRECT	Correction ON/OFF	[s][t]	1, 0	
1 PORT FULL CAL	OPEN	Open	[t]	-	
	SHORT	Short	[t]	-	
	LOAD	Load	[t]	-	
	DONE1PORT (DONE)	1 Port Full Cal Done	[t]	-	
2 PORT FULL CAL	S11OPEN	S11 Open	[t]	-	Note 1
	S11SHORT	S11 Short	[t]	-	Note 1
	S11LOAD	S11 Load	[t]	-	Note 1
	S22OPEN	S22 Open	[t]	-	Note 1
	S22SHORT	S22 Short	[t]	-	Note 1
	S22LOAD	S22 Load	[t]	-	Note 1
	DONEREFL	Reflection Done	[t]	-	Note 1
	FWDTRNS	Forward Transfer	[t]	-	Note 1
	FWDMATCH	Forward Match	[t]	-	Note 1
	REVTRNS	Reverse Transfer	[t]	-	Note 1
	REVMATCH	Reverse Match	[t]	-	Note 1
	DONETRNS	Transmission Done	[t]	-	Note 1
	OMITISO	Omit Isolation	[t]	-	Note 1
	FWDISO	Forward Isolation	[t]	-	Note 1
	REVISO	Reverse Isolation	[t]	-	Note 1
	DONEISO	Isolation Done	[t]	-	Note 1
DONE2PORT	2 Port Full Cal Done	[t]	-	Note 1	
CLEAR CAL DATA	CLEAR	Clear Cal Done	[t]	-	
CAL INTERPOLATION	INTERPOL	Interpolation on/off	[s][t]	1, 0	
CAL COPY CH1 → CH2	CCOPY	Cal Copy	[t]	1, 0	

Note 1 : Enable for type A only

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

Item	Code	Contents	Descriptive format	Response format	Remarks
CAL KIT	CKIT0	Don't care	[t]	1, 0	
	CKIT1	N 50Ω	[t]	1, 0	
	CKIT2	N 75Ω	[t]	1, 0	
	CKIT3	3.5mm	[t]	1, 0	
	CKIT4	7mm	[t]	1, 0	
PORT 1 FEMAL	PORT1FEM	Port 1 Femal	[t]	1, 0	Note 1
PORT 1 MAL	PORT1MAL	Port 1 Mal	[t]	1, 0	Note 1
PORT 2 FEMAL	PORT2FEM	Port 2 Femal	[t]	1, 0	Note 1
PORT 2 MAL	PORT2MAL	Port 2 Mal	[t]	1, 0	Note 1
ELECTRICAL LENGTH	LENGTH	on/off	[s][t]	1, 0	
	LENGVAL	Value	[d][u][t]	D(1)	

Note 1 : Enable for type A only

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

Item	Code	Contents	Descriptive format	Response format	Remarks
● MKR/ Δ MKR See chapter 3 of this manual Part 1 for description of basic functions.					
MARKER NUMBER	MKR1A	Marker # 1	[d][u][t]	D(s,r,i,lc)	*30
	MKR2A	Marker # 2	[d][u][t]	D(s,r,i,lc)	*30
	MKR3A	Marker # 3	[d][u][t]	D(s,r,i,lc)	*30
	MKR4A	Marker # 4	[d][u][t]	D(s,r,i,lc)	*30
	MKR5A	Marker # 5	[d][u][t]	D(s,r,i,lc)	*30
	MKR6A	Marker # 6	[d][u][t]	D(s,r,i,lc)	*30
	MKR7A	Marker # 7	[d][u][t]	D(s,r,i,lc)	*30
	MKR8A	Marker # 8	[d][u][t]	D(s,r,i,lc)	*30
	MKR9A	Marker # 9	[d][u][t]	D(s,r,i,lc)	*30
	MKR10A	Marker #10	[d][u][t]	D(s,r,i,lc)	*30

*30 : For MKR1A? to MKR10A?, the number of data item which is returned depending on the measuring condition at that time is different.

Parameter conversion		FORMAT	SMITH		POLAR	When other than SMITH and POLAR
		SMITH	LOG MKR	R+jX		
			Re/Im MKR	G+jB		
OFF			D(s,r,i)	D(s,r,i,lc)	D(s,r,i)	D(s,r)
ON	DEFAULT MKR		D(s,r,i)		D(s,r,i)	D(s,r)
	LIN MKR Re/Im MKR	LorC OFF	D(s,r,i)		D(s,r,i)	D(s,r,i)
		LorC ON	D(s,r,i,lc)		D(s,r,i,lc)	D(s,r,i,lc)

If the marker of specified marker is the active marker and Δ mode, no data on lc returns.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

Item	Code	Contents	Descriptive format	Response format	Remarks
MARKER OFF	MKRAOF	Marker all off	[t]	1, 0	
	MKROFF	Active marker off	[t]	
	MKR1OF	Marker # 1 off	[t]	1, 0	
	MKR2OF	Marker # 2 off	[t]	1, 0	
	MKR3OF	Marker # 3 off	[t]	1, 0	
	MKR4OF	Marker # 4 off	[t]	1, 0	
	MKR5OF	Marker # 5 off	[t]	1, 0	
	MKR6OF	Marker # 6 off	[t]	1, 0	
	MKR7OF	Marker # 7 off	[t]	1, 0	
	MKR8OF	Marker # 8 off	[t]	1, 0	
	MKR9OF	Marker # 9 off	[t]	1, 0	
	MKR10OF	Marker #10 off	[t]	1, 0	
MARKER TO MEM	MKRATOM	All to memory	[t]	*13
	MKRTOM	Active marker to memory	[t]	*13
MARKER TO DATA	MKRATOD	All to data	[t]	
	MKRTOD	Active marker to data	[t]	
COMPENSATE	MKRCMP	Compensate	[t]	1, 0	
	MKRUCMP	Uncompensate	[t]	1, 0	
COUPLE	MKRCOUP	Coupled	[t]	1, 0	
	MKRUCOUP	Uncoupled	[t]	1, 0	
SMITH MKR	SMKRLIN	Lin marker	[t]	1, 0	
	SMKRLOG	Log marker	[t]	1, 0	
	SMKRRI	Re/Im marker	[t]	1, 0	
	SMKRRX	R+jX marker	[t]	1, 0	
	SMKRGB	G+jB marker	[t]	1, 0	
POLAR MKR	PMKRLIN	Lin marker	[t]	1, 0	
	PMKRLOG	Log marker	[t]	1, 0	
	PMKRRI	Re/Im marker	[t]	1, 0	
IMPEDANCE MARKER	ZYMKDFLT	Default marker	[t]	1, 0	
	ZYMKLIN	Lin marker	[t]	1, 0	
	ZYMKRI	Re/Im marker	[t]	1, 0	
	ZYMKLC	LC on/off	{s}{t}	1, 0	
Smith Marker impedance Z0	MKRZ050	smith MKR Z0=50	[t]	1, 0	
	MKRZ075	smith MKR Z0=75	[t]	1, 0	

*13 : No execution unless in DISPDM mode

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

Item	Code	Contents	Descriptive format	Response format	Remarks
Δ REFERENCE	DMKRC	Δ REF= Δ MKR	[t] , [d][u][t]	1, 0	
	DMKRR	Δ REF=	[t] , [d][u][t]	1, 0	
		Δ REF. POSN			
	DMKRA	Active marker	[t] , [d][u][t]	1, 0	*14
	DMKRF	Δ REF=	[t] , [d][u][t]	1, 0	*15
		FIXED. MKR			
Δ MODE OFF	DMKROF	Δ mode off	[t]	1, 0	
FIXED MKR	FMKRS	Stimulus value	[t]	1, 0	*16
	FMKRV	Value	[d][u][t]	D(r)	*16
	MKRFIX	FIXED. MX→	[t]	*16
		ACT. M POSN.			
Δ RIPPLE	DRIPPL1	Δ ripple 1	[t]	D(r)	*17
	DRIPPL2	Δ ripple 2	[t]	D(r)	*17
	DLTX	Δ x	[d][u][t]	D(s)	
	DLTY	Δ y	[d][u][t]	D(r)	
	DMAXMIN	Δ max-min	[t]	D(r)	*17*18
	DRIPOFF	off	[t]	1, 0	
Δ'S OFFSET	DMKR10	Multi MKR Δ	[t] , [d][u][t]	1, 0	*19
	DMKR20	Multi MKR Δ	[t] , [d][u][t]	1, 0	*19
	DMKR30	Multi MKR Δ	[t] , [d][u][t]	1, 0	*19
	DMKR40	Multi MKR Δ	[t] , [d][u][t]	1, 0	*19
	DMKR50	Multi MKR Δ	[t] ; [d][u][t]	1, 0	*19
	DMKR60	Multi MKR Δ	[t] , [d][u][t]	1, 0	*19
	DMKR70	Multi MKR Δ	[t] , [d][u][t]	1, 0	*19
	DMKR80	Multi MKR Δ	[t] , [d][u][t]	1, 0	*19
	DMKR90	Multi MKR Δ	[t] , [d][u][t]	1, 0	*19
	DMKR100	Multi MKR Δ	[t] , [d][u][t]	1, 0	*19

- *14 : Because of delta between multimarkers, there is no execution unless several markers are ON.
- *15 : No execution unless fixed marker is ON.
- *16 : No execution unless format is in LOGMAG mode.
- *17 : No execution unless format is in LOGMAG or GDELAY mode.
- *18 : ON not possible unless in DMKRC or DMKRA mode
- *19 : Command for setting marker number which will serve as active marker in inter-multimarker delta mode.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

Item	Code	Contents	Descriptive format	Response format	Remarks
● MARKER SEARCH See chapter 3 of this manual Part 1 for description of basic functions.					
SEARCH	MAXSRCH	Max search	{t}	D(s, r, i, lc)	*24*31
	MINSRCH	Min search	{t}	D(s, r, i, lc)	*24*31
	SRCHOFF	Search off	{t}	
	LMAXSRC	Next max SRCH	{t}	D(s, r, i, lc)	*24*31
	LMINSRC	Next min SRCH	{t}	D(s, r, i, lc)	*24*31
TARGET	TREFMAX	Δ Ref. =max	{t}	1, 0	*16
	TREFREF	Δ Ref. =Ref	{t}	1, 0	*16
	TREFACT	Δ Ref. =Act MKR	{t}	1, 0	*16
	TREFCNT	Δ Ref. =C. F.	{t}	1, 0	*16
	T3DB	-3dB	{t}	D(s, r, s, s)	*29*16
	T6DB	-6dB	{t}	D(s, r, s, s)	*29*16
	T60DB	-60dB	{t}	D(s, r, s, s)	*29*16
	TXDB	-XdB	{d}{u}{t}	D(s, r, s, s)	*29*16
	TLEFT	Left Search	{t}	D(s, r)	*16
	TRIGHT	Right Search	{t}	D(s, r)	*16
	TIN	XdB down IN	{t}	1, 0	*16
	TOUT	XdB down OUT	{t}	1, 0	*16
FILTER ANALYSIS PHASE MKR	FLTANA	on/off	{s}{t}	1, 0	*16
	ZRPSRCH	Zero phase search	{t}	D(s, r)	*21
	TREFZRP	Δ Ref. =Zero search	{t}	1, 0	*21
	T3DEG	$\pm 3^\circ$	{t}	D(s, r)	*21
	T6DEG	$\pm 6^\circ$	{t}	D(s, r)	*21
	TXDEG	$\pm X^\circ$	{d}{u}{t}	D(s, r)	*21

- *16 : No execution unless format is in LOGMAG mode.
 - *21 : Cannot be executed when format is in PHASE or UNWRAP mode.
 - *24 : No valid data is returned if search command is not executed.
 - *29 : When FLTANA is OFF, D(s,r,s,s,).....(BW,Loss,f_L,f_R) is returned.
When FLTANA is ON, D(s,r,s,s,s,c,c,)..(BW,Loss,cf,Lf,Rf,Q,sf,) is returned.
 - *31 : For MAXSRCH? } The number of data item which is returned
MINSRCH? } depending on the measuring condition at
LMAXSRC? } that time is different.
LMINSRC? }
- The same as the table of *30.
However, for Δ mode, the data for lc is not returned.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

Item	Code	Contents	Descriptive format	Response format	Remarks
PART ANALYSIS	MKRPART	Part analysis	{s}{t}	1. 0	*18
TRACKING	MKRTRAC	Tracking	{s}{t}	1. 0	
<p>• MKR → See chapter 3 of this manual Part 1 for description of basic functions.</p>					
MKR →	MKRREF	MKR → Ref. value	{t}	
	MKRCENT	MKR → Center F.	{t}	*22
	MKRSTAR	MKR → Start F.	{t}	*22
	MKRSTOP	MKR → Stop F.	{t}	*22
	MKRSPAN	MKR → Span F.	{t}	*22
	MKRCACL	MKR → Center scale	{t}	
MARKER to MEMORY	MKR1TM	MKR # 1 to mem	{t}	1. 0	*13
	MKR2TM	MKR # 2 to mem	{t}	1. 0	*13
	MKR3TM	MKR # 3 to mem	{t}	1. 0	*13
	MKR4TM	MKR # 4 to mem	{t}	1. 0	*13
	MKR5TM	MKR # 5 to mem	{t}	1. 0	*13
	MKR6TM	MKR # 6 to mem	{t}	1. 0	*13
	MKR7TM	MKR # 7 to mem	{t}	1. 0	*13
	MKR8TM	MKR # 8 to mem	{t}	1. 0	*13
	MKR9TM	MKR # 9 to mem	{t}	1. 0	*13
	MKR10TM	MKR #10 to mem	{t}	1. 0	*13
MARKER to DATA	MKR1TD	MKR # 1 to data	{t}	1. 0	
	MKR2TD	MKR # 2 to data	{t}	1. 0	
	MKR3TD	MKR # 3 to data	{t}	1. 0	
	MKR4TD	MKR # 4 to data	{t}	1. 0	
	MKR5TD	MKR # 5 to data	{t}	1. 0	
	MKR6TD	MKR # 6 to data	{t}	1. 0	
	MKR7TD	MKR # 7 to data	{t}	1. 0	
	MKR8TD	MKR # 8 to data	{t}	1. 0	
	MKR9TD	MKR # 9 to data	{t}	1. 0	
	MKR10TD	MKR #10 to data	{t}	1. 0	
<p>• AUTO ZOOM</p>					
AUTO ZOOM	AUTOZOOM ATZMSPAN	AUTO ZOOM AUTO ZOOM SPAN	{t} {d}{u}{t} D(s)	*23*27

- *13 : No execution unless in DISPDM mode
- *18 : ON not possible unless in DMKRC or DMKRA mode
- *22 : MKR → Freq. when sweep type is LINFRQ
MKR → Level when sweep type is LEVEL sweep
- *23 : Cannot be executed when format is in LOGMAG or LINMAG mode.
- *27 : Not executed in SINGLE SWEEP, SWEEP HOLD, or EXTERNAL TRIGGER mode. After execution of this command, wait the next processing until SRQ of SWEEP END appears.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

Item	Code	Contents	Descriptive format	Response format	Remarks
Entry					
NUMERAL	0	0			
	1	1			
	2	2			
	3	3			
	4	4			
	6	6			
	7	7			
	9	9			
	.	.			
	-	-			
+	+				
EXP	EXP	EXP on ENT			
STEP	STPUP	↑			
	STPDN	↓			
	FU	⊙			
	CU	⊙			
	FD	⊙			
CD	⊙				
BACKSPACE	BS				
ENTRY OFF	EOFF				
UNITS	MHZ	MHz			
	KHZ	KHz			
	HZ	Hz			
	DEG	°			
	DP	dBm			
	DM	dBm			
	DB	dB			
	METER	m			
	CM	cm			

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

Item	Code	Contents	Descriptive format	Response format	Remarks
	SEC	sec			
	MSEC	msec			
	USEC	usec			
	NSEC	nsec			
	VOLT	V			
	MV	mV			
	UV	uV			
	NV	nV			
	UNIT	Unit			
	DIV	Div			
	PER, %	%			
DELIMITER	DLO		{t}	
	DL1		{t}	
	DL2		{t}	
	DL3		{t}	
IDENTIFI- CATION	IDNT	Identification	{t}	Strings	*1
INSTRUMENT PRESET	IP	Instrument preset	{t}	*28
<p>● PLOTTER See section 4.3 of this manual Part 1 for description of basic functions.</p>					
GPIB address	ADDRPLOT	Plotter GPIB address	{d}{u}{t}	D	
Plotter entry	PLT1PICT	Full size	{t}	1. 0	
	PLT2PICT	Half size	{t}	1. 0	
	PLT4PICT	Quarter size	{t}	1. 0	
	PLTEXEC	Execute	{t}	
	PLTABORT	Abort	{t}	
	PLT2LEFT	Half (LEFT)	{t}	1. 0	
	PLT2RIGHT	Half (RIGHT)	{t}	1. 0	
	PLT4LUP	Quarter (L, Up)	{t}	1. 0	
	PLT4LOW	Quarter (L, Lo)	{t}	1. 0	
	PLT4RUP	Quarter (R, Up)	{t}	1. 0	
PLT4RLOW	Quarter (R, Lo)	{t}	1. 0		

*1 : Response given as character string.

*28 : Insert a wait of 5-second after executing IP.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

Item	Code	Contents	Descriptive format	Response format	Remarks
Video Printer	PLTDATA	Data on/off	[s][t]	1, 0	Note 3
	PLTMEM	Memory on/off	[s][t]	1, 0	
	PLTMKR	Marker on/off	[s][t]	1, 0	
	PLTSCALE	Scale on/off	[s][t]	1, 0	
	PLTGRAT	Scaletype on/off	[s][t]	1, 0	
	PLTREFLN	Ref. line on/off	[s][t]	1, 0	
	PLTTEXT	Text all on/off	[s][t]	1, 0	
	PLTLABEL	Label on/off	[s][t]	1, 0	
	PLTD1PEN	PEN select CH1 data	[d][u][t]	D	
	PLTM1PEN	PEN select CH1 mem	[d][u][t]	D	
	PLTD2PEN	PEN select CH2 data	[d][u][t]	D	
	PLTM2PEN	PEN select CH2 mem	[d][u][t]	D	
	PLTSCLPEN	PEN select scale	[d][u][t]	D	
	PLTLBLPEN	PEN select label	[d][u][t]	D	
	PLTAT	PLOTOR type (AT)	[t]	1, 0	
	PLTHP	PLOTOR type (HP)	[t]	1, 0	
VPRINT	START/STOP Video Printer	[t]	-		
● SAVE/RECALL		See section 4.1 of this manual Part 1 for description of basic functions.			
SAVE/RECALL	SAVEREG1	Data save to reg1	[t]	-	
	SAVEREG2	Data save to reg2	[t]	-	
	SAVEREG3	Data save to reg3	[t]	-	
	SAVEREG4	Data save to reg4	[t]	-	
	SAVEREG5	Data save to reg5	[t]	-	
	SAVEREG6	Data save to reg6	[t]	-	
	SAVEREG7	Data save to reg7	[t]	-	
	SAVEREG8	Data save to reg8	[t]	-	
	SAVEREG9	Data save to reg9	[t]	-	
	SAVEREG10	Data save to reg10	[t]	-	
	RECLREG1	Data recall to reg1	[t]	1, 0	
	RECLREG2	Data recall to reg2	[t]	1, 0	
	RECLREG3	Data recall to reg3	[t]	1, 0	
	RECLREG4	Data recall to reg4	[t]	1, 0	
	RECLREG5	Data recall to reg5	[t]	1, 0	
	RECLREG6	Data recall to reg6	[t]	1, 0	
	RECLREG7	Data recall to reg7	[t]	1, 0	
	RECLREG8	Data recall to reg8	[t]	1, 0	
	RECLREG9	Data recall to reg9	[t]	1, 0	
RECLREG10	Data recall to reg10	[t]	1, 0		
RECLPOFF	Power off recall	[t]	1, 0		

Note 3 : Enable for R3761/3762

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

Item	Code	Contents	Descriptive format	Response format	Remarks
	CLRREG1	clear reg1	[t]	-	
	CLRREG2	clear reg2	[t]	-	
	CLRREG3	clear reg3	[t]	-	
	CLRREG4	clear reg4	[t]	-	
	CLRREG5	clear reg5	[t]	-	
	CLRREG6	clear reg6	[t]	-	
	CLRREG7	clear reg7	[t]	-	
	CLRREG8	clear reg8	[t]	-	
	CLRREG9	clear reg9	[t]	-	
	CLRREG10	clear reg10	[t]	-	
<p>● SAVE/RECALL(FILE) See section 4.1 of this manual Part 1 for description of basic functions.</p>					
LOAD FILE	UDFILE	LOAD FILE	[strings][t]	-	*8 *26
STORE FILE	STFILE1	STORE FILE	[strings][t]	-	*8 *26
DEFINE STORE	RAWARY	RAW DATA on/off	[s][t]	1, 0	
	CORARY	CORR DATA on/off	[s][t]	1, 0	
	DATAARY	DATA on/off	[s][t]	1, 0	
	MEMARY	MEM on/off	[s][t]	1, 0	
PURGE	PURGE	Purge	[strings][t]	-	*8 *26
INITIALIZE	INITIAL	Initialize	[t]		

- *8 : Append character string after GPIB code.
*26 : Always insert a sufficient wait period to ensure end of floppy disk access after executing this command.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

Item	Code	Contents	Descriptive format	Response format	Remarks
● SRQ					
	SRQE	SRQ, interrupt enable	[t]	1, 0	
	SRQD	SRQ, interrupt disable	[t]	1, 0	
● REAL TIME CLOCK See chapter 3 of this manual Part 1 for description of basic functions.					
REAL TIME CLOCK	RTC30ADJ	30sec ADJUST	[t]	-	
	YEAR	YEAR	[d][u][t]	D	*25
	MONTH	MONTH	[d][u][t]	D	*25
	DAY	DAY	[d][u][t]	D	*25
	WEEK	WEEK	[d][u][t]	D	*25
	HOUR	HOUR	[d][u][t]	D	*25
	MINUTE	MINUTE	[d][u][t]	D	*25
● SCREEN					
EDIT	EDIT	EDIT mode (on/off)	[s][t]	1, 0	*10
● Special Function					
SETTLING VARIABLE	SETLVARI	Settling Variable on/off	[s][t]	1, 0	
SETTLING TIME	SETLTIME	Settling time	[d][u][t]	D (t)	
MKRPOINT SOURCE CORRECTION	MKRPOINT SRCCOR	MKR Point on/off Source Linearity Correction on/off	[s][t] [s][t]	1, 0 1, 0	Note 3 Note 3

Note 3 : Enable for R3761/3762

*10 : Measuring menu set by EDITOFF, and EDITOR menu set by EDITON.

*25 : Always insert a wait of at least 1 second after executing this command.

*37 : As a result of execution, 0 to 6 are returned. The relation between these values and week is as follows.

0: Sunday	2: Tuesday	4: Thursday
1: Monday	3: Wednesday	5: Friday
		6: Saturday

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

Item	Code	Contents	Descriptive format	Response format	Remarks
INPUT CORRECTION	INPCOR	Input Frequency Response Correction on/off	[s][t]	1, 0	Note 3
LIMIT LINE	LIMILINE	LIMIT LINE on/off	[s][t]	1, 0	
	LIMITEST	LIMIT TEST on/off	[s][t]	1, 0	
	FAILBEEP	LIMIT FAIL BEEP on/off	[s][t]	1, 0	
	LIMCOMP	LIMIT Compensate on/off	[s][t]	1, 0	
	LIMISTIO	LIMIT STIMULUS offset	[d][u][t]	D	*35
	LIMIAMPO	LIMIT AMPLITUDE offset	[d][u][t]	D	*35
	LSEGCL	Clear All Segments	[t]	1, 0	
	LSEG	Segment No.	[d][u][t]	D	*35
	LSTIM	Segment Stimulus	[d][u][t]	D	*35
	LIMU	Segment Upper Limit	[d][u][t]	D	*35
	LIML	Segment Lower Limit	[d][u][t]	D	*35
	LIMTFLT	Flat Line Type	[t]	1, 0	
	LIMTSLP	Slope Line Type	[t]	1, 0	
	LIMTSP	Single Point Type	[t]	1, 0	

GPIB code	Contents	Remarks
● TRACE DATA (OUTPUT)		
OT1DRAT	CH1 input meas and raw data following AVG	
OT1MRAT	CH1 mem raw data	*33
OT2DRAT	CH2 input meas and raw data following AVG	
OT2MRAT	CH2 mem raw data	*33
OT1CORDI	CH1 directional error coefficient	*34
OT1CORSO	CH1 source match error coefficient	*34
OT1CORTR	CH1 tracking error coefficient	*34
OT2CORDI	CH2 directional error coefficient	*34
OT2CORSO	CH2 source match error coefficient	*34
OT2CORTR	CH2 tracking error coefficient	*34
OT2CORN	CH2 normalized averaging data	

Note 3 : Enable for R3761/3762

*33 : No input or output permitted if Mem is not ON.

*34 : No input or output permitted if correction is not ON.

*35 : Terminate is UNIT.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

GPIB code	Contents	Remarks
OT1CORED	CH1 data after error correction	
OT2CORED	CH2 data after error correction	
OT2CORNRR	CH2 normalized averaging data	
OT1NORED	CH1 data after data/mem operation	
OT2NORED	CH2 data after data/mem operation	
OT1DFOR	CH1 data after formatting	*32
OT1MFOR	CH1 mem after formatting	*32 *33
OT2DFOR	CH2 data after formatting	*32
OT2MFOR	CH2 mem after formatting	*32 *33

*32 : Formatted Data and Mem are input/output by radian unit in PHASE and UNWRAPPED PHASE. For additional format, they are input or output as data that is the same as that on screen display.

*33 : No input or output permitted if Mem is not ON.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

Item	Code	Contents	Descriptive format	Response format	Remarks
● SPECIAL FUNC					
SETTLING VARIABLE	SETLVARI	Settling variable on/off	[s] [t]	1, 0	
SETTLING TIME	SETLTIME	Settling time	[d] [u] [t]	D (t)	
● TRACE DATA (FORMAT)					
Trace data format	FORM0	ASCII	[t]	----	
	FORM2	IEEE 32 bit, binary	[t]	----	
	FORM3	IEEE 64 bit, binary	[t]	----	
	FORM5	IEEE 32 bit, binary, byte sequence rearrangement	[t]	----	
	FORM6	IEEE 64 bit, binary, byte sequence rearrangement	[t]	----	
	FORM7	N88 BASIC 32 bit, binary	[t]	----	
	FORM8	N88 BASIC 64 bit, binary	[t]	----	
	Output of measurement data point count	OTMP		[t]	----

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

GPIB code	Contents	Remarks
● TRACE DATA (INPUT)		
IN1DRAT	CH1 input meas and raw data following AVG	
IN1MRAT	CH1 mem raw data	*33
IN2DRAT	CH2 input meas and raw data following AVG	
IN2MRAT	CH2 mem raw data	*33
IN1CORDI	CH1 directional error coefficient	*34
IN1CORSO	CH1 source match error coefficient	*34
IN1CORTR	CH1 tracking error coefficient	*34
IN2CORDI	CH2 directional error coefficient	*34
IN2CORSO	CH2 source match error coefficient	*34
IN2CORTR	CH2 tracking error coefficient	*34
IN1CORN	CH1 normalized averaging data	*34
IN2CORN	CH2 normalized averaging data	*34
IN1CORED	CH1 data after error correction	
IN2CORED	CH2 data after error correction	
IN1NORED	CH1 data after data/mem operation	
IN2NORED	CH2 data after data/mem operation	
IN1DFOR	CH1 data after formatting	
IN1MFOR	CH1 mem after formatting	*33
IN2DFOR	CH2 data after formatting	
IN2MFOR	CH2 mem after formatting	*33

*33 : No input or output permitted if Mem is not ON.

*34 : No input or output permitted if correction is not ON.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.4 GPIB Input and Output Formats

Table 2 - 2 GPIB Program Code (cont'd)

GPIB code	Contents	Remarks
● LIMIT LINE (OUTPUT)		
OT1LIMF	CH1 limit test result outputs information on the fail point.	*36
OT2LIMF	CH2 limit test result outputs information on the fail point.	*36
OT1LIML	CH1 outputs information on the limit test result of all limit test points.	*36
OT2LIML	CH2 outputs information on the limit test result of all limit test points.	*36
OT1LIMM	CH1 outputs information on the limit test result of the MKR point.	*36
OT2LIMM	CH2 outputs information on the limit test result of the MKR point.	*36

*36 : Format is common, and is listed in LIMIT LINE (OUTPUT).

2.5 Service Request

The status register is outlined in Figure 2-1 below.

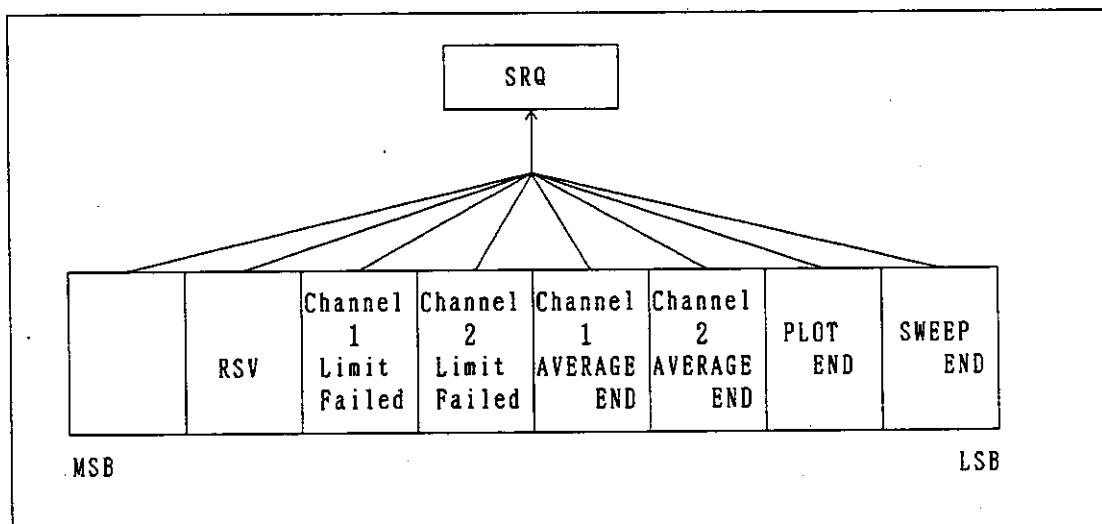


Figure 2 - 1 Status Register

Note : If the serial pole is done for this instrument, 1 is always set to the RSV.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

2.6 Program Examples

2.6.1 Program for Determining Difference between Very Large and Very Small Points within Same Specified Frequencies, and Maximum Value of Difference Between Adjacent Inflection Points

To run this program, set the GPIB address to 11 with the network analyzer in TALKER/LISTENER mode.

HP200 Series

```

100  !
110  !      SAMPLE PROGRAM
120  !
130  OUTPUT 711;"CH1 ARIN LOGMAG"
140  OUTPUT 711;"SPANF      12 MHZ"
150  OUTPUT 711;"CENTERF   57 MHZ"
160  OUTPUT 711;"SDIV      10 DB"
170  OUTPUT 711;"REFV       0 DB"
180  OUTPUT 711;"REFF      100 PER"
190  OUTPUT 711;"OUTLEV     0 DB"
200  OUTPUT 711;"AI50AO    RBW1KHZ"
210  OUTPUT 711;"M301P"
220  OUTPUT 711;"MKRCMPON"
230  OUTPUT 711;"LINFREQ"
240  OUTPUT 711;"MKR1A     53 MHZ"
250  OUTPUT 711;"DMKRC"
260  OUTPUT 711;"MKR1A      9 MHZ"
270  OUTPUT 711;"DLTX      40 KHZ"
280  OUTPUT 711;"DLTY     0.01 DB"
290  OUTPUT 711;"DRIPPL1"
300  OUTPUT 711;"DRIPPL1?"
310  ENTER  711;Rip11
320  OUTPUT 711;"DRIPPL2"
330  OUTPUT 711;"DRIPPL2?"
340  ENTER  711;Rip12
350  PRINT  Rip11,Rip12
360  END

```

PC9800 Series

```

100  '
110  '      SAMPLE PROGRAM
120  '
130  PRINT @11;"CH1 ARIN LOGMAG"
140  PRINT @11;"SPANF      12 MHZ"
150  PRINT @11;"CENTERF   57 MHZ"
160  PRINT @11;"SDIV      10 DB"
170  PRINT @11;"REFV       0 DB"
180  PRINT @11;"REFF      100 PER"
190  PRINT @11;"OUTLEV     0 DB"
200  PRINT @11;"AI50AO    RBW1KHZ"
210  PRINT @11;"M301P"
220  PRINT @11;"MKRCMPON"
230  PRINT @11;"LINFREQ"
240  PRINT @11;"MKR1A     53 MHZ"
250  PRINT @11;"DMKRC"
260  PRINT @11;"MKR1A      9 MHZ"
270  PRINT @11;"DLTX      40 KHZ"
280  PRINT @11;"DLTY     0.01 DB"
290  PRINT @11;"DRIPPL1"
300  PRINT @11;"DRIPPL1?"
310  INPUT @11;RIPL1
320  PRINT @11;"DRIPPL2"
330  PRINT @11;"DRIPPL2?"
340  INPUT @11;RIPL2
350  PRINT RIPL1,RIPL2
360  STOP

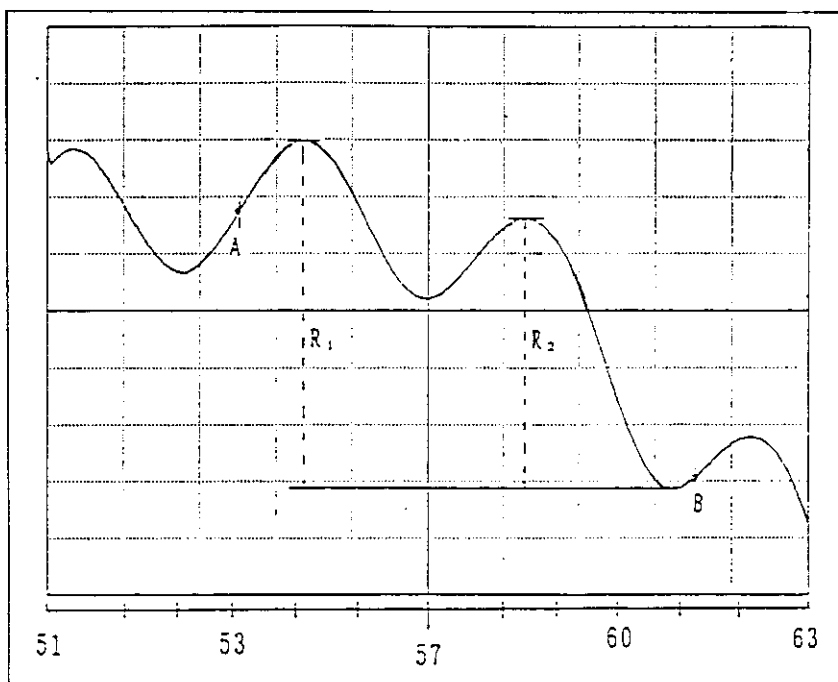
```

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

Commentary

Address	Contents
130	Channel 1 INPUT A/R LOGMAG
140	SPAN 12MHz
150	CENTER 57MHz
160	/DIV Set to 10 dB
170	REF LEVEL Set to 0 dB
180	REF Position Set to 100%
190	OUTPUT LEVEL Set to 0 dB
200	Impedance 50 ohms
	Attenuator 0 dB
	RESOLUTION band width 1 kHz
210	Set to measuring point 301
220	MARKER COMPENSATE mode ON
230	Linear sweep
240) Set point A
250	
260	Set (plus B point) OFFSET 9 MHz at point A
270	Differential coefficient (ΔX)
300) Read RIPPLE1 from Network Analyzer
310	
320	Compute RIPPLE2 (R_2)
330) Read RIPPLE2 from Network Analyzer
340	
350	Display
360	End



NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

2.6.2 Trace Data Input/Output

● TRACE DATA (INPUT)

To run this program, set the GPIB address to 11 with the network analyzer in TALKER/LISTENER mode.

HP200 Series

PC9800 Series

```

100 DIM R(600)
110 Add=711
120 OUTPUT Add;"M601P"
130 OUTPUT Add;"IN1DFOR"
140 FOR I=0 TO 600
150   OUTPUT Add;R(I)
160   OUTPUT Add;Imag
170 NEXT I
180 OUTPUT Add;"TREND"
190 END
  
```

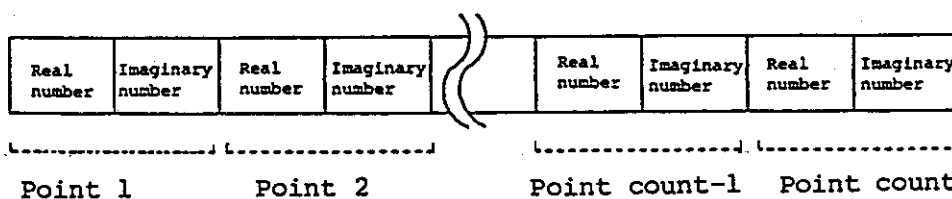
```

100 DIM R(600)
110 ADD=11
120 PRINT @ADD;"M601P"
130 PRINT @ADD;"IN1DFOR"
140 FOR I=0 TO 600
150   PRINT @ADD;R(I)
160   PRINT @ADD;Imag
170 NEXT I
180 PRINT @ADD;"TREND"
190 STOP
  
```

Commentary

Address	Contents
100	Array declaration
110	GPIB address setting
120	Specify measuring points as 601 points
130	Request input of TRACE DATA
140	Loop for the number of points
150	Data output to the network analyzer (real number)
160	Data output to the network analyzer (imaginary number : dummy output when not required)
170	
180	End of data output to the network analyzer
190	End

Note : TRACE DATA is input in real/imaginary number sequence at each point.



NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

- Data in excess of the network analyzer measurement points is disregarded.

For example, if the network analyzer measurement point is set to 601 and data consisting of 602 or more points is sent to the network analyzer, the excess points are disregarded.

- The "TREND" in line 180 must always be input when the transfer is completed.
- TRACE DATA (OUTPUT)

To run this program, set the GPIB address to 11 with the network analyzer in TALKER/LISTENER mode.

HP200 Series

PC9800 Series

```

100 DIM R(1200)
110 Add=711
120 OUTPUT Add;"OTIDFOR"
130 ENTER Add;Po
140 FOR I=0 TO Po-1
150   ENTER Add;R(I)
160   ENTER Add;Imag
170 NEXT I
180 PRINT R(*)
190 END

```

```

100 DIM R(1200)
110 ADD=11
120 PRINT @ADD;"OTIDFOR"
130 INPUT @ADD;PO
140 FOR I=0 TO PO-1
150   INPUT @ADD;R(I)
160   INPUT @ADD;IMAG
170   PRINT R(I)
180 NEXT I
190 STOP

```

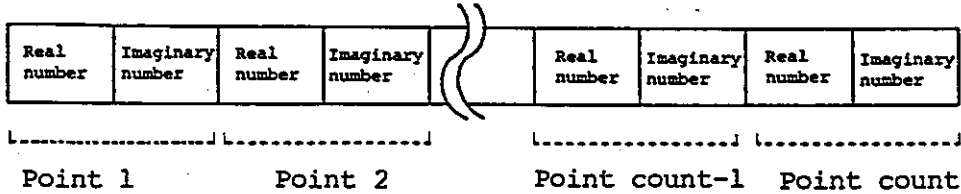
Commentary

Address	Contents
100	Array declaration
110	GPIB address setting
120	Request output of TRACE DATA
130	Enter the number of points
140	Loop for the number of points
150	Data input (real number)
160	Data input (imaginary number : dummy output when not required)
170	Output (only for PC9800 Series)
180	Output (only for HP200 Series)
190	end

Note : For output of TRACE DATA, the number of points is output at first, then output in order of real number, and imaginary number at each point, as shown below.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples



2.6.3 Limit Line Output

● LIMIT LINE (OUTPUT)

To run this program, set the GPIB address to 11 with the network analyzer in TALKER/LISTENER mode.

HP200 Series

PC9800 Series

```

100 DIM ST(1200)
110 DIM RE(1200)
120 DIM UP(1200)
130 DIM LO(1200)
140 Add=711
150 OUTPUT Add;"OT1LIMF"
160 ENTER Add;Po
170 FOR I=0 TO Po-1
180 ENTER Add;ST(I),RE(I),UP(I),LO(I)
190 NEXT I
200 END
  
```

```

100 DIM ST(1200)
110 DIM RE(1200)
120 DIM UP(1200)
130 DIM LO(1200)
140 ADD=11
150 PRINT @ADD;"OT1LIMF"
160 INPUT @ADD;PO
170 FOR I=0 TO PO-1
180 INPUT @ADD;ST(I),RE(I),UP(I),LO(I)
190 NEXT I
200 STOP
  
```

Commentary

Address	Contents
100	} Array declaration
130	
140	
150	
160	Request output of LIMIT LINE
160	Enter the number of points
170	Loop for the number of points
180	Data input (in order of stimulus, decision result, upper-limit value, and lower-limit value)
190	
200	End

Note : For output of LIMIT LINE, the number of points is output at first, then output in order of stimulus value, decision result, upper-limit value, and lower-limit value at each point.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

Stimulus value	Decision result	Upper-limit value	Lower-limit value		Stimulus value	Decision result	Upper-limit value	Lower-limit value
Point 1				//	Point Po			

As a result of decision, -1 indicates no-decision and 1 indicates PASS.

2.6.4 SRQ

When the GPIB code "SRQE" is executed, the sweep end SRQ output is passed to the external controller.

To run this program, set the GPIB address to 11 with the network analyzer in TALKER/LISTENER mode.

HP200 Series

PC9800 Series

```

10  OUTPUT 711;"SRQE"
20  ON INTR GOSUB 100
30  ENABLE INTR
40  ' LOOP
50  GOTO 40
100 ' SWEEP END
110 S=SPOL(711)
120 IF S (<) 65 THEN GOTO 170
130 OUTPUT 711;"MAXSRCH"
140 OUTPUT 711;"MAXSRCH?"
150 ENTER 711 ;F, R, I, LC
160 PRINT R
170 RETURN
180 STOP

```

```

10  PRINT @11;"SRQE"
20  ON ISRQ GOSUB 100
30  SRQ ON
40  ' LOOP
50  GOTO 40
100 ' SWEEP END
110 POLL 11, S
120 IF S <> 65 THEN 170
130 PRINT @11;"MAXSRCH"
140 PRINT @11;"MAXSRCH?"
150 INPUT @11;F, R, I, LC
160 PRINT R
170 RETURN
180 STOP

```

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

Commentary

Address	Contents
10	The network analyzer sweep end SRQ output designation
20	Branch to line number 100 when SRQ arrives
30	Interrupt enable
40	Loop
50	
100	
110	Serial pole
120	Go to 170 if other than sweep end
130	Retrieval of maximum value with marker
140	Request for outputting the value indicated by marker
150	Marker-indicated value read
160	Print level display
170	Return
180	End

- Example of sweep program with SRQ

Built-in BASIC

```
100 OUTPUT 31;"SINGLE"  
110 GOSUB *SWP  
.  
.  
.  
300 *SWP  
310 ON ISRQ GOTO *PATH  
320 OUTPUT 31;"SRQE"  
330 ENABLE INTR  
340 OUTPUT 31;"SINGLE"  
350 *LOOP  
360 GOTO *LOOP  
370 !  
380 *PATH  
390 DISABLE INTR  
400 OUTPUT 31;"SRQD"  
410 RETURN  
.  
.  
.
```

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

Commentary

Address	Contents
100	Set address to SINGLE, and stop sweep (This is to receive the SRQ securely).
110	Divided into subroutine *SWP.
310	Specify the destination when receiving SRQ.
320	Output GPIB code 'SRQE'. When 'SRQE' is executed, sweep is terminated and SRQ is output at the same time.
330	Permit interrupt (SRQ is enabled in this case).
340	Sweep once.
360	Loop
390	Inhibit interrupt (SRQ is disabled in this case).
400	Output GPIB code 'SRQD'. When 'SRQE' is executed, sweep is terminated and no SRQ is output.

This program is a basic sweep type. The program has the part that can be deleted.

2.6.5 Starting BASIC from External Controller

While the network analyzer is in TALKER/LISTENER mode, BASIC commands can be executed from the external controller.

"@BASIC command"

Appending @ to the beginning enables BASIC commands inside the network analyzer to be activated from the external controller.

Description of Program Example

Certain BASIC programs are generated in advance in built-in BASIC, and saved to the network analyzer floppy disk under file names "FILE_1", "FILE_2", "FILE_3", and "FILE_4". Then when program example 1 is generated and executed by external controller, program in the network analyzer is loaded and run one after another.

- Note:
- To run these programs, set the GPIB address to 11 with the network analyzer in TALKER/LISTENER mode.
 - The built-in BASIC REQUEST command has been included to inform the external controller when execution is completed.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

Program example 1

HP200 Series

```
100 DIM A$(3) [6] , L$ [20]
110 F=4
115 E=0
120 ON INTR 7 GOSUB 1000
130 A$(0)="FILE_1"
140 A$(1)="FILE_2"
150 A$(2)="FILE_3"
160 A$(3)="FILE_4"
200 FOR I=0 TO F-1
205 M$=CHR$(34)&A$(I)&CHR$(34)
210 L$="@LOAD "&M$
220 OUTPUT 711;L$
230 WAIT 5
240 OUTPUT 711;"@RUN"
250 ENABLE INTR 7;2
260 IF E=0 THEN 260
270 WAIT .5
280 E=0
290 NEXT I
1000 ! SRQ
1010 S=SPOLL(711)
1020 IF S=65 THEN
1030     BEEP
1040     E=1
1050     END IF
1060 RETURN
1070 END
```

PC9800 Series

```
100 DIM A$(3)
110 F=4
115 E=0
120 ON SRQ GOSUB 1000
130 A$(0)="FILE1"
140 A$(1)="FILE2"
150 A$(2)="FILE3"
160 A$(3)="FILE4"
200 FOR I=0 TO F-1
205 M$=CHR$(34)+A$(I)+CHR$(34)
210 L$="@LOAD "+M$
220 PRINT @11;L$
230 FOR J=1 TO 30000 : NEXT J
240 PRINT @11;"@RUN"
250 SRQ ON
260 IF E=0 THEN 260
270 FOR J=1 TO 3000 : NEXT J
280 E=0
290 NEXT I
1000 'SRQ
1010 POLL 11.S
1020 IF S=65 THEN BEEP:E=1
1060 RETURN
1070 STOP
```

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

● Built-in BASIC

FILE_1

```
100 FOR I=1 TO 24
110 PRINT I
120 NEXT I
130 REQUEST 64+1
```

Commentary

Address	Contents
100	Loop 24 times
110	Display I
120	
130	REQUEST to HOST

FILE_2

```
100 FOR I=1 TO 24
110 PRINT I*2
120 NEXT I
130 REQUEST 64+1
```

Address	Contents
100	Loop 24 times
110	Display I*2
120	
130	REQUEST to HOST

FILE_3

```
100 FOR I=24 TO 1 STEP -1
110 PRINT I
120 NEXT I
130 REQUEST 64+1
```

Address	Contents
100	Loop 24 times (minus steps)
110	Display I
120	
130	REQUEST to HOST

FILE_4

```
100 FOR I=24 TO 1 STEP -1
110 PRINT "ADVANTEST NETWORK ANALYZER"
120 NEXT I
130 REQUEST 64+1
```

Commentary

Address	Contents
100	Loop 24 times (minus steps)
110	Display ADVANTEST NETWORK ANALYZER
120	
130	REQUEST to HOST

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

2.6.6 Data Sending/Receiving with SRQ

If BASIC is started in [2.6.3 SRQ] and [2.6.4 External controller], data can be sent or received by running the subprogram for the network analyzer.

- Setting equipment
 PC9801 : Controller
 Network analyzer : Talker/listener

● Outline of program

PC9801 side	Network Analyzer side
1. Start the network analyzer program.	→ 1. Initialize the network analyzer.
2. Wait for SRQ (SWEEP END request statement).	2. Set center frequency and frequency bandwidth.
3. Perform serial pole (wait for 255 from the request statement).	← 3. Sweep once. Output SRQ after terminating sweep.
4. Receive data.	4. Measurement (using the built-in function)
5. Return to 1 or terminate the program.	← 5. Output SRQ (using the request statement).
	← 6. Send data.
	7. Termination

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

PC9800 series

Network Analyzer (built-in BASIC)

```

100 '*****
110 '***PC9801 - V.N.A DATA TRANSFER ***
120 '***                                     ***
130 '*****
140 '
150 '
160 NA=11
170 PRINT @NA;"SRQD"
180 ON SRQ GOSUB *SR
190 PRINT @NA;"@RUN"
200 SRQ ON
210 FLG=0
220 IF FLG=1 THEN GOTO *RECEIVE
230 GOTO 220
240 '
250 '
260 *SR
270  POLL NA,P
280  IF P=255 THEN FLG=1
290  RETURN
300 '
310 '
320 *RECEIVE
330  INPUT @NA;F
340  INPUT @NA;LEVEL
350  PRINT "F=";F;" (HZ)"
360  PRINT "LEVEL=";LEVEL;" (dB)"
370 '
380  INPUT "NEXT DATA ? (yes:ret/no:other)";Q$
390  IF Q$="" THEN 160
400  STOP

```

```

100 !*****
110 !***PC9801-V.N.A DATA TRANSFER ***
120 !***                                     ***
130 !***                                     ***
140 !*****
150 !
160 !*** INITIALIZE NA ***
170 !
180 OUTPUT 31;"IP"
190 BUZZER 0,3000
200 OUTPUT 31;"SPANF 1 MHZ"
210 OUTPUT 31;"CENTERF 100 MHZ"
220 !
230 !ONE SWEEP
240 !
250 ON ISRQ GOTO *SEND
260 OUTPUT 31;"SINGLE"
270 OUTPUT 31;"SRQE"
280 ENABLE INTR
290   *LOOP
300   GOTO *LOOP
310 !
320 *SEND
330 OUTPUT 31;"SRQD"
340 DISABLE INTR
350 !
360 F=FREQ(600,0)
370 LEVEL=VALUE(600,0)
380 !
390 REQUEST 255
400 OUTPUT 11;F
410 OUTPUT 11;LEVEL
420 !
430 STOP

```

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

< Commentary >

PC9801 Side

Address	Contents
160	Network analyzer=11 (Network analyzer address)
170	Set the network analyzer to the SRQ OFF mode.
180	Jump to *SR after receiving SRQ.
190	Start the subprogram for the network analyzer.
200	Permit interrupt (start to receive SRQ).
210	FLG=0
220	Jump to *RECEIVE if FLG equals to 1.
230	To line 220
260	*SR
270	Perform serial pole.
280	Set FLG to 1 if the serial pole result is 255.
290	Return to line 190.
320	*RECEIVE
330	Receive frequency data.
340	Receive amplitude data.
350	Display frequency data.
360	Display amplitude data.
380	Select whether re-measurement is done.
390	Jump to line 160 if re-measurement is done.
400	Termination

Network Analyzer Side

Address	Contents
180	Initialize the network analyzer.
190	Wait for three seconds.
200	Set frequency bandwidth.
210	Set center frequency.
250	Jump to *SEND after receiving SRQ.
260	Set SWEEP to the SINGLE mode.
270	Set the network analyzer to the SRQ ON mode.
280	Permit interrupt (start to receive SRQ).
290	*LOOP
300	GOTO*LOOP
320	*SEND
330	Set the network analyzer to the SRQ OFF mode.
340	Inhibit interrupt (SRQ receive inhibition).
360	Measure frequency data
370	Measure amplitude data.
390	Output SRQ (using the request statement).
400	Send frequency data.
410	Send amplitude data.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

2.6.7 Program Example Using External Controller or Built-in Basic

When using an external controller

To run this program, set the GPIB address to 11 with the network analyzer in TALKER/LISTENER mode.

HP200 Series

PC9800 Series

100	OUTPUT 711; "EDITOFF "	100	PRINT @11;"EDITOFF"
110	OUTPUT 711; "LOGMAG"	110	PRINT @11;"LOGMAG"
120	OUTPUT 711; "CENTERF100MHZ"	120	PRINT @11;"CENTERF100HZ"
130	OUTPUT 711; "SPANF10MHZ"	130	PRINT @11;"SPANF10MHZ"
140	OUTPUT 711; "AUTO"	140	PRINT @11;"AUTO"
150	OUTPUT 711; "CENTERF ?"	150	PRINT @11;"CENTERF?"
160	ENTER 711;Cf	160	INPUT @11;CF
170	OUTPUT 711; "SPANF ?"	170	PRINT @11;"SPANF?"
180	ENTER 711;Sf	180	INPUT @11;SF
190	OUTPUT 711; "MAXSRCH "	190	PRINT @11;"MAXSRCH"
200	OUTPUT 711; "MAXSRCH ?"	200	PRINT @11;"MAXSRCH?"
210	ENTER 711;F, L, D1, D2	210	INPUT @11;F, L, D1, D2
220	PRINT "Center freq. = ", Cf	220	PRINT @11;"Center freq. =", CF
230	PRINT "Span freq. = ", Sf	230	PRINT @11;"Span freq. =", SF
240	PRINT "MAX Level = ", L	240	PRINT @11;"MAX Level =", L
250	END	250	STOP

<Commentary>

Address	Contents
100	Switch to measurement menu
110	LOGMAG mode
120	Set central frequency to 100 MHz
130	Set frequency bandwidth to 10 MHz
140	Execute auto scale
150	Request center frequency response
160	Substitute center frequency response in variable Cf
170	Request frequency bandwidth response
180	Substitute frequency bandwidth response in variable Sf
190	Search for maximum level
200	Request maximum level response
210	Substitute maximum level response in each variable
220	Display center frequency
230	Display frequency bandwidth
240	Display maximum level
250	

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

When using built-in BASIC

(When the built-in BASIC is used, the network analyzer itself can be controlled specifying OUTPUT and ENTER address as 31.)

```

100 OUTPUT 31; "EDITOFF "
110 OUTPUT 31; "LOGMAG"
120 OUTPUT 31; "CENTERF100MHZ "
130 OUTPUT 31; "SPANF10MHZ"
140 OUTPUT 31; "AUTO"
150 OUTPUT 31; "CENTERF ?"
160 ENTER 31;Cf
170 OUTPUT 31; "SPANF ?"
180 ENTER 31;Sf
190 OUTPUT 31; "MAXSRCH "
200 OUTPUT 31; "MAXSRCH ?"
210 ENTER 31;F, L, D1, D2
220 PRINT "Center freq. = " , Cf
230 PRINT "Span  freq. = " , Sf
240 PRINT "MAX  Level = " , L
250 STOP

```

<Commentary>

Address	Contents
100	Switch to measurement menu
110	LOGMAG mode
120	Set central frequency to 100 MHz
130	Set frequency bandwidth to 10 MHz
140	Execute auto scale
150	Request center frequency response
160	Substitute center frequency response in variable Cf
170	Request frequency bandwidth response
180	Substitute frequency bandwidth response in variable Sf
190	Search for maximum level
200	Request maximum level response
210	Substitute maximum level response in each variable
220	Display center frequency
230	Display frequency bandwidth
240	Display maximum level
250	

2.6.8 X'TAL Filter Measuring Program Example

```
1000 REM -----
1100 REM SAMPLE PROGRAM FOR
1200 REM XTAL FILTER
1300 REM
1400 REM -----
1500 REM FILTER IS . . .
1600 REM 21.4MHz BPF
1700 REM -----
1800 REM
1900 REM
2000 REM *** INITIALIZE NA ***
2100 REM
2200 OUTPUT 31; "CH1 ARIN LOGMAG "
2300 OUTPUT 31; "SDIV 10 DB"
2400 OUTPUT 31; "REFV 0 DB "
2500 OUTPUT 31; "REFP 100 PER"
2600 OUTPUT 31; "REFLON PORT2"
2700 OUTPUT 31; "OUTLEV 0 DB "
2800 OUTPUT 31; "AI50A20 "
2900 OUTPUT 31; "RBW1KHZ "
3000 OUTPUT 31; "FREE CGNT M301P "
3100 OUTPUT 31; "MKRCMP"
3200 REM
3300 REM *** LOOP TOP ***
3400 REM
3500 OUTPUT 31; "SPANF 25 KHZ"
3600 OUTPUT 31; "CENTERF 21.4 MHZ"
3700 REM
3800 REM *** 1 SWEEP ***
3900 REM
4000 OUTPUT 31; "SINGLE"
4100 BUZZER 0 1500
4200 REM
4300 REM *** SCREEN INITIALIZE ***
4400 REM
4500 CLS
4600 FOR I=1 TO 10
4700 PRINT
4800 NEXT I
4900 REM
5000 REM *** GET INS LOSS ***
5100 REM
5200 LOSS=MAX(0,1200,0)
5300 MAXP=PMAX(0,1200,0)
5400 PRINT "LOSS", LOSS, "dB"
5500 REM
```

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

(cont'd)

```
5600 REM *** GET RIPPLE ***
5700 REM
5800 RIPPLE=RPL1(400,800,4,0.01,0)
5900 PRINT "RIPPLE",RIPPLE,"dB"
6000 REM
6100 REM *** GET BW (3dB) ***
6200 REM
6300 BW3DB=BND(600,3,0)
6400 PRINT "BW (3dB)",BW3DB,"Hz"
6500 REM
6600 REM *** GET BW (40DB) ***
6700 REM
6800 BW40DB=BND(600,40,0)
6900 PRINT "BW (40dB)",BW40DB,"Hz"
7000 REM
7100 REM *** 1MHz DEVIATION LEVEL ***
7200 REM
7300 OUTPUT 31; "SPANF 2 MHZ"
7400 OUTPUT 31; "SINGLE"
7500 BUZZER 0 1500
7600 LLEVEL=VALUE(0,0)
7700 RLEVEL=VALUE(1200,0)
7800 PRINT "1MHz DEV. LEVEL(dB)"
7900 PRINT LLEVEL,RLEVEL
8000 GOTO 3200
8100 REM
8200 REM *** END JOB ***
8300 REM
8400 OUTPUT 31; "CONT"
8500 END
```

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

< Comentary >

Address	Contents
2000	Initialization
to	
4200	
5000	Measure insertion loss
to	
5500	
5600	Ripple measurement
to	
6000	
6100	Measure 3 dB bandwidth
to	
6500	
6600	Measure 40 dB dandwidth
to	
7000	
7100	Measure levels at ± 1 MHz away from tuned frequency
to	
7700	
8000	Return to loop top and repeat measurement

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

2.6.9 Example of Measuring Program Using Parallel I/O Ports

```
1000 REM *****
1010 REM ***                               ***
1020 REM ***   NETWORK ANALYZER           ***
1030 REM ***                               ***
1040 REM ***   SEMI AUTO PROGRAM BY PIO   ***
1050 REM ***                               ***
1060 REM *****
1070 REM
1080 CURSOR 0 18
1090 PRINT "*** NA DEMO PROGRAM *** "
1100 PRINT ""
1110 PRINT " * USE PIO DEMO SET"
1120 PRINT
1130 PRINT "[1] NALLOW BAND TEST"
1140 PRINT "[2] WIDE BAND TEST"
1150 PRINT "[3] PHASE MEASUREMENT"
1160 PRINT "[4] G.D. MEASUREMENT"
1170 PRINT ""
1180 OUTPUT 31; "CHI ARIN LOGMAG"
1190 OUTPUT 31; "SDIV 10 DB"
1200 OUTPUT 31; "REFV 0 DB"
1210 OUTPUT 31; "REFP 100 PER"
1220 OUTPUT 31; "REFLON PORT2"
1230 OUTPUT 31; "OUTLEV 0 DB"
1240 OUTPUT 31; "B11A20"
1250 OUTPUT 31; "A11A20"
1260 OUTPUT 31; "R150A20"
1270 OUTPUT 31; "RBW1KHZ"
1280 OUTPUT 31; "FREE CONT M301P"
1290 OUTPUT 31; "MKRCMP"
1300 OUTPUT 31; "SPLITON"
1310 OUTPUT 31; "DUALOFF"
1320 OUTPUT 31; "CENTERF 455 KHZ"
1330 BUZZER 0 1000
1340 CURSOR 2. 28
1350 *LOOPTOP
1360 CURSOR 2. 28
1370 PRINT "SELECT PIO NUMBER ?"
1380 *LOOPTOP1
1390 ENTER 32;PIO
1400 IF PIO=1 THEN GOTO *MEAS1
1410 IF PIO=2 THEN GOTO *MEAS2
1420 IF PIO=4 THEN GOTO *MEAS3
1430 IF PIO=8 THEN GOTO *MEAS4
```

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

(cont'd)

```
1440 GOTO *LOOPTOP1
1450 REM
1460 REM
1470 REM
1480 REM--- NALLOW BAND MEASURE ---
1490 *MEAS1
1500 CLS
1510 OUTPUT 31; "SPANF 100 KHZ"
1520 OUTPUT 31; "LOGMAG"
1530 REM
1540 REM *** 1 SWEEP ***
1550 REM
1560 CURSOR 0,19
1570 BUZZER 0 1000
1580 CLS
1590 REM
1600 REM *** SCREEN INITIALIZE ***
1610 REM
1620 CURSOR 0,19
1630 REM
1640 REM *** GET INS LOSS ***
1650 REM
1660 LOSS=MAX(0,1200,0)
1670 MAXP=PMAX(0,1200,0)
1680 PRINT "LOSS",LOSS,"dB"
1690 REM
1700 REM *** GET RIPPLE ***
1710 REM
1720 RIPPLE=RPL1(400,800,4,0.01,0)
1730 PRINT "RIPPLE",RIPPLE,"dB"
1740 REM
1750 REM *** GET BW(83dB) ***
1760 REM
1770 BW3DB=BND(600,3,0)
1780 PRINT "BW (3dB)",BW3DB,"Hz"
1790 REM
1800 REM *** GET BW (40DB) ***
1810 REM
1820 BW40DB=BND(600,40,0)
1830 PRINT "BW (40dB)",BW40DB,"Hz"
1840 GOTO *LOOPTOP
1850 REM
1860 REM ---WIDE BAND MEASUREMENT ---
1870 REM
1880 *MEAS2
1890 CLS
```

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

(cont'd)

```
1900 OUTPUT 31; "SPANF 2 MHZ"
1910 OUTPUT 31; "LOGMAG"
1920 BUZZER 0 1000
1930 CURSOR 0.19
1940 CLS
1950 LLEVEL=VALUE(0.0)
1960 RLEVEL=VALUE(1200.0)
1970 CLS : CURSOR 0 20
1980 PRINT "1MHz DEV. LEVEL (dB) "
1990 PRINT LLEVEL,RLEVEL
2000 GOTO *LOOPTOP
2010 REM
2020 REM
2030 REM
2040 END
2050 REM --- PHASE MEASUREMENT ---
2060 REM
2070 *MEAS3
2080 CLS
2090 OUTPUT 31; "SPANF 100 KHZ"
2100 OUTPUT 31; "PHASE"
2110 REM
2120 REM *** 1SWEEP ***
2130 REM
2140 CURSOR 0 19
2150 CLS
2160 REM
2170 REM *** SCREEN INITIALIZE ***
2180 REM
2190 CURSOR 0 19
2200 GOTO *LOOPTOP
2210 REM
2220 REM --- DELAY NMEASUREMENT ---
2230 REM
2240 *MEAS4
2250 CLS
2260 OUTPUT 31; "SPANF 100 KHZ"
2270 OUTPUT 31; "DELAY"
2280 BUZZER 0 3000
2290 OUTPUT 31; "AUTO"
2300 REM
2310 REM *** 1 SWEEP ***
2320 REM
2330 CURSOR 0 19
2340 BUZZER 0 2000
2350 GOTO *LOOPTOP
```


NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

Comentary

Address	Contents
1180	Initialization
to	
1330	
1370	Set measuring function by parallel I/O input signal
to	
1430	
1480	Return to loop top and repeat measurement
to	MEAS1 measurement on basis of narrow band frequency span
1840	(Insertion loss, ripple, 3 dB bandwidth, 40 dB bandwidth
1860	MEAS2 measurement
to	Measure levels of start and stop points on basis of wide
2000	band frequency span
2050	MEAS3 measurement
to	Phase measurement
2200	Return to loop top and repeat measurement
2240	MEAS4 measurement
	Group delay measurements
2350	Return to loop top and repeat measurement

2.6.10 Example of Program Where Limited Test Function Is Used in Low-pass
Filter Measurements

```
1000  !
1010  !
1020  ! INITIALIZE
1030  !
1040  OUTPUT 31: "CH1 LOGMAG"
1050  OUTPUT 31: "MKRCMP"
1060  OUTPUT 31: "SINGLE"
1070  OUTPUT 31: "STARTF 1.5MHZ"
1080  OUTPUT 31: "STOPF 6 MHZ"
1090  OUTPUT 31: "DUAL ON"
1100  OUTPUT 31: "SPLIT ON"
1110  OUTPUT 31: "COUPLE ON"
1120  OUTPUT 31: "CH2 DELAY"
1130  BUZZER 0 500
1140  OUTPUT 31: "SRQE"
1150  !
1160  ! MEASUREMENT
1170  !
1180  BUZZER 4 100
1190  OUTPUT 31: "MEAS"
1200  ON ISRQ GOTO 1240
1210  ENABLE INTR
1220  !
1230  GOTO 1220
1240  !
1250  Fr=FMIN(0.1200,0)
1260  F1=MIN(0.1200,0)
1270  F2=POINT1(2e+06,0)
1280  L2=VALUE(F2,0)
1290  F3=POINT1(3e+06,0)
1300  L2=VALUE(F3,0)
1310  F4=POINT1(4e+06,0)
1320  L4=VALUE(F4,0)
1330  Fi=POINT1(3.58e+06,0)
1340  Li=VALUE(Fi,0)
1350  !
1360  ! DELAY
1370  !
1380  BUZZER 0 500
1390  F3=POINT1(3.58e+06,1)
1400  D3=VALUE(F3,1)
1410  F3=POINT1(4.08e+06,1)
1420  D4=VALUE(F4,1)
1430  !
```

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

(cont'd)

```

1440 ! GO/NOTO CHECK !!
1450 !
1460 CURSOR 0,3
1470 N1=LMTUL1(Fr,5.3025e+06,4.7975e+06)
1480 N2=LMTUL1(F1,-30,-200)
1490 N3=LMTUL1(L2,-5,-11)
1500 N4=LMTUL1(L3,5,-1.2)
1510 N5=LMTUL1(L4,5,-1.2)
1520 N6=LMTUL1(L1,5,-1)
1530 N7=LMTUL1(D3,230,170)
1540 N8=LMTUL1(D4,330,0)
1550 N=N1+N2+N3+N4+N5+N6+N7+N8
1560 IF N=0 THEN GOTO 1590
1570 PRINT "NG !!"
1580 GOTO 1180
1590 PRINT "OK !!"
1600 GOTO 1180
1610 STOP

```

< Comentary >

Address	Contents
1020	Initialization
to	
1120	
1130	500 msec wait
1140	Enable SRQ
1200	Set internal SRQ interrupt and branch
1210	Accept interrupt
1250	Measured value interrupt at frequency measurement point
to	specified by CH1
1340	
1380	Measured value interrupt at frequency measurement point
to	specified by CH2
1420	
1470	Designation of limit values for each measured value
to	
1540	
1550	Set branching according to result of comparison value
1570	Print NG if even a single item was NG
1590	Print OK if all items are OK, and continue to measure repeatedly

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

```
10  REM -----
20  REM
30  REM      XTAL EQUIVALENT CIRCUITM
40  REM
50  REM
60  REM      PI-CIRCUIT-METHOD
70  REM
80  REM -----
90  REM
100 REM
110 REM
120 REM
130 REM
140 REM
150 REM
160 REM
170 REM
180 SPAN1$ = "SPANF 1KHZ"
190 CENTER1$ = "CENTER 11.97596430MHZ"
200 CLS : CURSOR 0 14
210 REM
220 REM -----
230 REM
240 REM      START
250 REM
260 REM -----
270 NA=31
280 CFLAG=0
320 OUTPUT NA; "COUPLEON"
330 PRINT
340 PRINT
350 PRINT "Do you need CAL?   YES;1 NO;0 "
360 INPUT QQ
380 IF QQ=1 THEN CFLAG=1
390 GOTO *MEAS
400 *CALUC
410 REM
420 REM *** CALUCLATE ***
430 REM
440 XDEG=3
450 RR=25*(10 ^ (-LOSS/20)-1)
460 AA=1+0.50878*(RR/12.5)
470 BB=2*0.50878*(RR/12.5)
480 CC=FR*PI*2*XDEG
490 DD=180*DF3
500 Q=(AA/BB)*(CC/DD)
```

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

(cont'd)

```

510 C1=1/(2*PI*FS*RP*Q)
520 L=1/((2*PI*FS) ^ 2*C1)
530 PRINT "**** NA DEMO (XTAL) **** "
540 PRINT "LOSS (dB) " , -LOSS
550 PRINT "Fs (Hz) " , FS
560 PRINT "Fr (Hz) " , FR
570 PRINT "dF (Hz) " , DF3
580 PRINT
590 PRINT "Q " , Q
600 PRINT "Rr (ohm) " , RR
610 PRINT "C1 (pF) " , C1*1e+12
620 PRINT "L (mH) " , L*1000
630 PRINT "-----"
640 GOTO *MEAS2
650 REM
660 REM *** MEASUREMENT ***
670 REM
680 *MEAS
690 OUTPUT NA; "DUALON"
700 OUTPUT NA; "SPLITOFF"
710 FOR CH=1 TO 2
720     IF CH=1 THEN GOTO 750
730     OUTPUT NA; "CH2"
740     GOTO *EX1
750     OUTPUT NA; "CH1"
760     *EX1
770     OUTPUT NA;SPANIS
780     OUTPUT NA;CENTERIS
790     OUTPUT NA; "ARIN"
800     OUTPUT NA; "PORT2"
810     OUTPUT NA; "A150A0"
820     OUTPUT NA; "B150A20"
830     OUTPUT NA; "R050A20"
840     OUTPUT NA; "RBW30HZ"
850     OUTPUT NA; "MKRCMP"
860     OUTPUT NA; "STIME 0.1 SEC"
870     OUTPUT NA; "M101P"
880     OUTPUT NA; "FREE CONT"
890 NEXT CH
900 OUTPUT NA; "CH1 LOGMAG"
910 OUTPUT NA; "REFV 0 DB"
920 OUTPUT NA; "REFP 90 PER"
930 OUTPUT NA; "CH2 PHASE"
940 OUTPUT NA; "REFV 0 DEG"
950 OUTPUT NA; "REFP 50 PER"
960 OUTPUT NA; "SINGLE"
970 REM
980 REM *** CALIBRATION ***
990 REM
1000 *CAL

```

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

(cont'd)

```
1010 IF CFLAG=0 THEN GOTO *MEAS2
1020 OUTPUT NA; "CH1 NORMOFF"
1030 OUTPUT NA; "CH2 NORMOFF"
1040 CLS
1050 BEEP
1060 PRINT ">> CONNECT (THRU) "
1070 INPUT "& PRESS (RETURN) KEY" ,Q$
1080 PRINT "Calibration....."
1090 BUZZER 0 3000
1100 OUTPUT NA; "CH1 NORMON"
1110 OUTPUT NA; "CH2 NORMON"
1120 PRINT "CAL done."
1130 BEEP
1140 PRINT ">> CONNECT (DUT)
1150 INPUT "& PRESS (RETURN) KEY" ,Q$
1160 PRINT "MEASURING START "
1170 REM
1180 REM *** MEASURE START ***
1190 REM
1200 *MEAS2
1210 OUTPUT NA; "SRQE"
1220 OUTPUT NA; "MEAS"
1230 ON ISRQ GOTO 1260
1240 ENABLE INTR
1250 GOTO 1240
1260 REM
1270 REM *** GET MAG DATA ****
1280 REM
1290 OUTPUT NA; "CH1 "
1300 LOSS=MAX(0,1200,0)
1310 FS=FMAX(0,1200,0)
1320 REM
1330 REM *** GET PHASE DATA ***
1340 REM
1350 OUTPUT NA; "CH2"
1360 OUTPUT NA; "ZRPSRCH"
1370 OUTPUT NA; "MKRIA?"
1380 ENTER NA;FR
1390 OUTPUT NA; "TREFZRP"
1400 OUTPUT NA; "T3DEG"
1410 OUTPUT NA; "T3DEG?"
1420 ENTER NA;DF3
1430 REM
1440 REM
1450 CLS
1460 GOTO *CALUC
1470 REM
1480 REM
1490 END
```

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

<Comentary>

Address	Contents
180	Set center frequency to 11.97596430 MHz, and spanwidth to
to	1kHz
190	
200	Clear screen, and decide the cursor position
290	Switch marker couple ON
to	
320	
350	Select whether CAL is required or not (0 or 1)
to	
370	
390	Jump to initialization routine
420	X'TAL element constant calculation and display of result
440	Calculate X'TAL element constant
to	
520	
530	Display result of X'TAL element constant calculation
to	
630	
650	Initialize the network analyzer
to	
660	
680	Switch dual-channel display ON, and split display OFF
to	
700	
710	Form loop required to set two channels CH1 and CH2
to	
750	
770	Various setting conditions
to	
960	
970	Calibration routine
to	
980	
1000	Determine whether calibration is necessary, then proceed to
to	initialization
1030	
1040	Clear screen display
1060	Display short bar connection message
to	
1080	
1100	Proceed with normalization
to	
1120	
1130	Display [DUT] ... X'TAL connection message
to	
1160	

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

<Comentary >

Address	Contents
1180 to 1240	Routine for repeating sweep, and output/detection of service request at end of sweep
1270 to 1310	Built-in function for return of maximum amplitude level and corresponding frequency in screen display during amplitude measurement mode
1340 to 1420	Return of value of frequency 3 dB bandwidth for phase value of 0° in phase measurement mode
1460	Jump to calculation routine

2.6.11 AUTO SCALE

Note the following items to perform AUTO SCALE.

Built-in BASIC

```
100 OUTPUT 31;"DUALON"  
110 OUTPUT 31;"CENTERF 235 MHZ"  
120 OUTPUT 31;"CH2 DELAY"  
130 OUTPUT 31;"SPANF 10 MHZ"  
140 GOSUB *SWP  
150 OUTPUT 31;"CH2 AUTO"  
160 OUTPUT 31;"CH1 AUTO"  
.  
.  
.  
300 *SWP  
310 ON ISRQ GOTO *PATH  
320 OUTPUT 31;"SRQE"  
330 ENABLE INTR  
340 OUTPUT 31;"SINGLE"  
350 *LOOP  
360 GOTO *LOOP  
370 !  
380 *PATH  
390 DISABLE INTR  
400 OUTPUT 31;"SRQD"  
410 RETURN  
.  
.  
.
```

When this program is run, AUTO SCALE is performed. If it is run except line 140, no AUTO SCALE is performed completely. Because sweep will stop temporarily to screen setting at line 100 to 130. To prevent sweep from stopping temporarily, the program is swept with subroutine sweeping once.

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

2.6.12 Binary Data Input and Output

(1) Example of N88 BASIC trace data read-out program (64 bit, binary)

```

10 ' TRACE DATA NA->PC example program
20 ' "FORM8" N88 double precision binary mode
30 OPTION BASE 1
40 DIM X$(1201,2)
50 ISET IFC:ISET REN
60 CMD DELIM=0 ' delimiter of PC is CR/LF
70 CMD TIMEOUT=0
80 NA=11 ' GPIB address of NA
90 PRINT @NA;"OTLDFOR"
100 PRINT @NA;"OTMP"
110 INPUT @NA;MP ' read No. of data points
120 PRINT @NA;"FORM8" ' double precision binary mode
130 FOR I=1 TO MP
140 LINE INPUT @NA;X$
150 X$(I,1)=CVD(X$) ' real part is first 8 bytes
160 X$(I,2)=CVD(MID$(X$,9)) ' imag part is second 8 bytes
170 NEXT I
180 END

```

Comentary

Address	Contents
90	Specifies data transmitted from the network analyzer as display data
100	Requests output of display data point count
110	Reads display data point count
120	Specifies the N88 BASIC internal double precision, floating point format as the network analyzer output format
130	Reads data and converts it internally
170	

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

(2) Example of N88 BASIC trace data write program

```

10 ' TRACE DATA PC->NA example program
20 ' "FORM8" N88 double precision binary mode
30 OPTION BASE 1
40 DIM X#(1201,2)
50 ISET IFC:ISET REN
60 CMD DELIM=0 ' delimiter of PC is CR/LF
70 CMD TIMEOUT=0
80 NA=11 ' GPIB address of NA
90 PRINT @NA;"OT1DFOR"
100 PRINT @NA;"OTMP"
110 INPUT @NA;MP ' read No. of data points
120 PRINT @NA;"FORM8" ' double precision binary mode
130 PRINT @NA;"IN1DFOR"
140 CMD DELIM=3 ' delimiter of PC is EOI
150 FOR I=1 TO MP
160 X$=MKD$(X#(I,1))+MKD$(X#(I,2))
170 PRINT @NA;X$ @ ' send real and imag
180 NEXT I
190 END

```

<Comentary>

Address	Contents
90	Specifies the display data for reading display data point count (Does not read the trace data actually)
100	Requests output of display data point count
110	Reads display data point count
120	Specifies the N88 BASIC internal double precision, floating point format as the network analyzer input format
130	Enables the network analyzer to input data
140	Specifies PC output format
150	Reads data and converts it internally
}	
180	

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

(3) Example of N88 BASIC trace data read program (32 bit, binary)

```

10 ' TRACE DATA NA->PC example program
20 ' "FORM7" N88 single precision binary mode
30 OPTION BASE 1
40 DIM X(1201,2)
50 ISET IFC:ISET REN
60 CMD DELIM=0 ' delimiter of PC is CR/LF
70 CMD TIMEOUT=0
80 NA=11 ' GPIB address of NA
90 PRINT @NA;"OT1DFOR"
100 PRINT @NA;"OTMP"
110 INPUT @NA;MP ' read No. of data points
120 PRINT @NA;"FORM7" ' single precision binary mode
130 FOR I=1 TO MP
140 LINE INPUT @NA;X$
150 X(I,1)=CVS(X$) ' real part is first 4 bytes
160 X(I,2)=CVS(MID$(X$,5)) ' imag part is second 4 bytes
170 NEXT I
180 END

```

<Comentary>

Address	Contents
90	Specifies data transmitted from the network analyzer as display data
100	Requests output of display data point count
110	Reads display data point count
120	Specifies the N88 BASIC internal single precision, floating point format as the network analyzer output format
130	Reads data and converts it internally
}	
170	

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

(4) Example of N88 BASIC trace data write program (32 bit, binary)

```

10 ' TRACE DATA PC->NA example program
20 ' "FORM7" N88 single precision binary mode
30 OPTION BASE 1
40 DIM X(1201,2)
50 ISET IFC:ISET REN
60 CMD DELIM=0 ' delimiter of PC is CR/LF
70 CMD TIMEOUT=0
80 NA=11 ' GPIB address of NA
90 PRINT @NA;"OT1DFOR"
100 PRINT @NA;"OTMP"
110 INPUT @NA;MP ' read No. of data points
120 PRINT @NA;"FORM7" ' single precision binary mode
130 PRINT @NA;"IN1DFOR"
140 CMD DELIM=3 ' delimiter of PC is EOI
150 FOR I=1 TO MP
160 X$=MKS$(X(I,1))+MKS$(X(I,2))
170 PRINT @NA;X$ e ' send real and imag
180 NEXT I
190 END

```

<Comentary>

Address	Contents
90	Specifies the display data write program (32 bit, binary) (Does not read trace data actually)
100	Requests output of display data point count
110	Reads display data point count
120	Specifies the N88 BASIC internal single precision, floating point format as the network analyzer input format
130	Enables the network analyzer to input data
140	Specifies PC output delimiter
150	Write data
}	
180	

NETWORK ANALYZER
PROGRAMMING MANUAL

2.6 Program Examples

(5) Example of HP200/300 series 64-bit binary data read program

```
10 ' TRACE DATA NA->HP300,200 series example program
20 ' "FORM3" IEEE 64 bit floating
30 REAL X(0:1200,0:1) BUFFER
40 INTEGER Na,Mp,N,I
50 Na=711
60 ASSIGN @Na TO Na
70 OUTPUT @Na;"FORM3"
80 OUTPUT @Na;"OT1DFOR"
90 OUTPUT @Na;"OTMP"
100 ENTER @Na;Mp
110 N=Mp*8*2
120 ASSIGN @Buf TO BUFFER X(*)
130 TRANSFER @Na TO @Buf;COUNT N,WAIT
140 END
```

<Comentary >

Address	Contents
70	Specifies 64-bit binary (IEEE) mode
80	Specifies display data reading
90	Requests output of display data point count
100	Reads display data point count
110	Calculates the number of bytes transmitted
120,130	Transmits binary data

NETWORK ANALYZER
PROGRAMMING MANUAL

3.1 Outline

3. CONTROL MODE

3.1 Outline

The network analyzer is equipped with a GPIB controller function capable of controlling external equipment. By using the BASIC programming function, both the network analyzer itself and external equipment connected to the network analyzer can be controlled.

— CAUTION —



If the GPIB is locked when in controller mode, press the network analyzer STOP key three times to initialize the GPIB port.

NETWORK ANALYZER
PROGRAMMING MANUAL

3.2 Setting Controller Mode

3.2 Setting Controller Mode

Select the system controller function by pressing the front panel LOCAL

switch and selecting  from the softmenu. Then select  and

key in the network analyzer's GPIB address (0 thru 30) by pressing the corresponding numeric keys. Addressing is also necessary when setting controller mode.

CAUTION

- The GPIB address of external equipment connected to the network analyzer must not be the same as the network analyzer address.
- The address specified at this stage is used for internal processing purposes. The address used for controlling the network analyzer by built-in BASIC program is fixed to "31".

3.3 Handling Floppy Disks

(1) Floppy Disk Dimensions and Component Parts

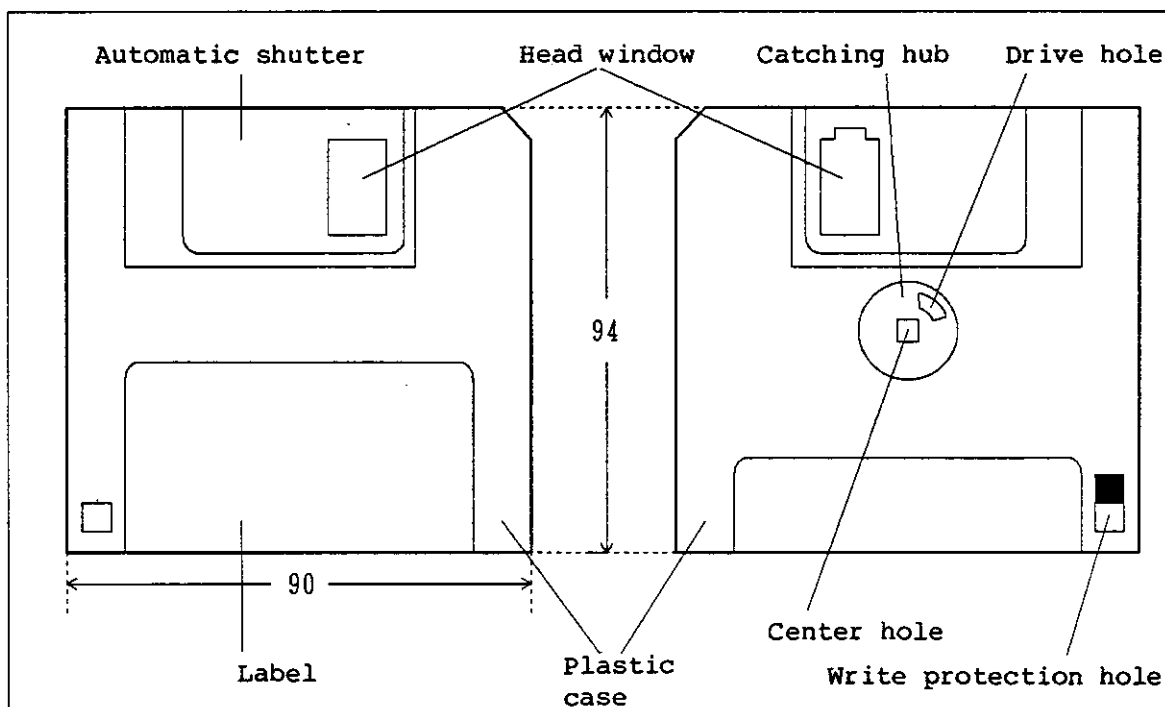


Figure 3 - 1 Floppy Disk Dimensions and Component Parts

- Label : The label is affixed by the user when a floppy disk is used.
- Head window : Head window apertures are located on both sides of the disk at the same position as the read/write heads. The heads move vertically across these apertures. When a floppy disk is removed from the drive slot, the automatic shutter closes to protect the disk surface.
- Catching hub (drive and center holes) : When a floppy disk is inserted into the drive slot, it is secured and rotated by a spindle using a catching magnet.
- Write protection hole : This hole prevents important data from being erased accidentally by operational error.

NETWORK ANALYZER
PROGRAMMING MANUAL

3.3 Handling Floppy Disks

Take note of the following precautions when storing floppy disks after removal from the drive.

- ① Keep floppy disks away from magnetic fields and other strong magnetic materials.
- ② Protect floppy disks from heat sources and direct sunlight.
- ③ Heat, cigarette ash, and other foreign matter can also lead to floppy disk damage.
- ④ Do not touch the magnetically coated surface by hand, and do not try to clean the surface by hand.
- ⑤ Do not place heavy articles on the top of floppy disks.

If the floppy disk is damaged (wetted, folded, bent) or stained by foreign matter, it must be replaced with the new one. If such disk is used, the drive head will become dirty and cannot be used. The other floppy disk may also become dirty.

CAUTION

The floppy disk contents may not be read correctly if the power is switched on with a disk already mounted in the drive. In this case, switch the power off, and remove the floppy disk before switching the power back on.

(3) Write Protect

To prevent valuable data from being erased accidentally by operational error etc., writing additional data to that disk can be inhibited by the write protect feature.

Write protect is enabled by write protect notch (Figure 3-3). Normally, this knob is left in the position nearest the center hole to permit writing, but is moved to the corner position to prevent writing.

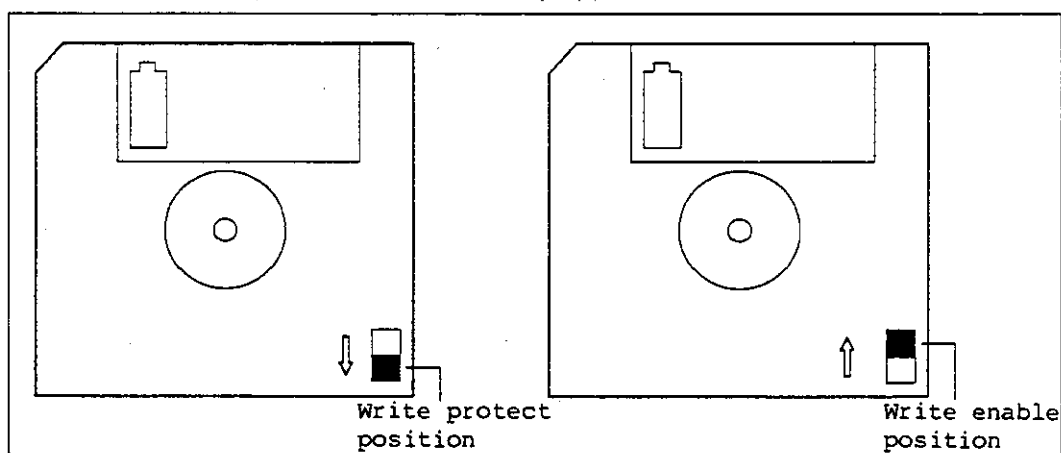


Figure 3 - 3 Floppy Disk Write Protect and Write Enable

NETWORK ANALYZER
PROGRAMMING MANUAL

3.4 File Management

3.4 File Management

3.4.1 Outline

BASIC programs, saved data, and other information stored on floppy disk are called "files". Files can be displayed, erased, and copied.

The main factors involved in storage of information on floppy disks are briefly described below.

DISKNAME : To identify individual floppy disks, DISKNAME is written when the disk is initialized. (See to initialization in section 3.4.3.)

FILE : Basic programs, save data, and other information are stored in individual files which may take up any number of sectors.

SECTOR : The smallest unit in which data can be stored on floppy disk.
1 sector corresponds to 512 bytes.

File type : File groups are separated into three types; BASIC, SYSTEM, and DATA. Data is a clear statements so that still more few types of file exist.

Disk capacity : The maximum data storage capacity per disk is as follows:

Maximum number of files : 200
Total number of sectors : 1400

Data can be stored as long as neither of these limits is exceeded.

3.4.2 Saving and Recalling Programs

Generated programs will be lost when the power is switched off if they are not stored on floppy disk.

The SAVE command is used to store programs. And the LOAD command is used to recall programs from floppy disk.

By using various save/recall functions of the network analyzer, saved data can also be recorded as files on floppy disk.

3.4.3 Initialization of Floppy Disk

Before a floppy disk can be used in the network analyzer, it must first be initialized by writing data of predetermined format to that disk.

Note, however, that when a used disk is initialized, all previous data stored on the disk is lost: Therefore, before initializing a disk, always check its contents. Disk directory information can be checked by using CAT or CHKDSK.

Floppy disks are initialized by using the INITIALIZE command.

Example:

INITIALIZE) ... ADVANTEST:NA and disk name are determined automatically.

INITIALIZE "DEMO.DISK" ... The character string enclosed between double quotation marks (") becomes the disk name.

CAUTION

The disk name can consist of up to 16 characters, the available characters being the same as those used in file names.

(Refer to **NOTE** of item 24. SAVE in section 5.3.)

3.4.4 File Management

CAT and CHKDSK

The CAT command is used to display the directory of the currently inserted disk. Directory details include (reading from left to right) registration number, file name, number of sectors used, number of characters, and file attributes.

The CHKDSK command is used to display disk information such as the disk name registered when the disk was initialized, number of files, and number of disk sectors used.

3.4.5 File Storage

SAVE "File Name"

The SAVE command is used to store programs on floppy disk after appending a file name to the program. If a file name which already exists on that disk is specified, the contents of that file are updated.

3.4.6 File Recalling

LOAD "File Name"

The LOAD command is used to retrieve files from floppy disk to memory.

3.4.7 File Deletion

PURGE "File Name"

The PURGE command is used to remove unwanted files.

3.4.8 File Name Change

RENAME "Old File Name", "New File Name"

The RENAME command is used to change the name of current files without changing their contents.

CAUTION

File names can consists of up to 16 characters except alphanumeric characters and double quotation mark (").

MEMO



A large, empty rectangular box with rounded corners, intended for writing the memo's content.

NETWORK ANALYZER
PROGRAMMING MANUAL

4.1 Outline

4. BASIC PROGRAMMING

4.1 Outline

In addition to general purpose BASIC commands, the BASIC language incorporated in the network analyzer is also equipped with GPIB control commands and the network analyzer dedicated built-in functions. Small-scale GPIB systems can be readily constructed.

The programming area capacity is 192 kbytes, generally allowing programming of more than 2,000 program steps.

4.2 Activation of Program Mode

(1) Program Mode

Program mode can be activated by pressing the PROGRAM key on the network analyzer front panel, or by pressing CHG MODE on the keyboard. As a result, the display shown below appears on the CRT screen. Since this is a toggle key, program mode is switched back to measuring mode if the key is pressed again.

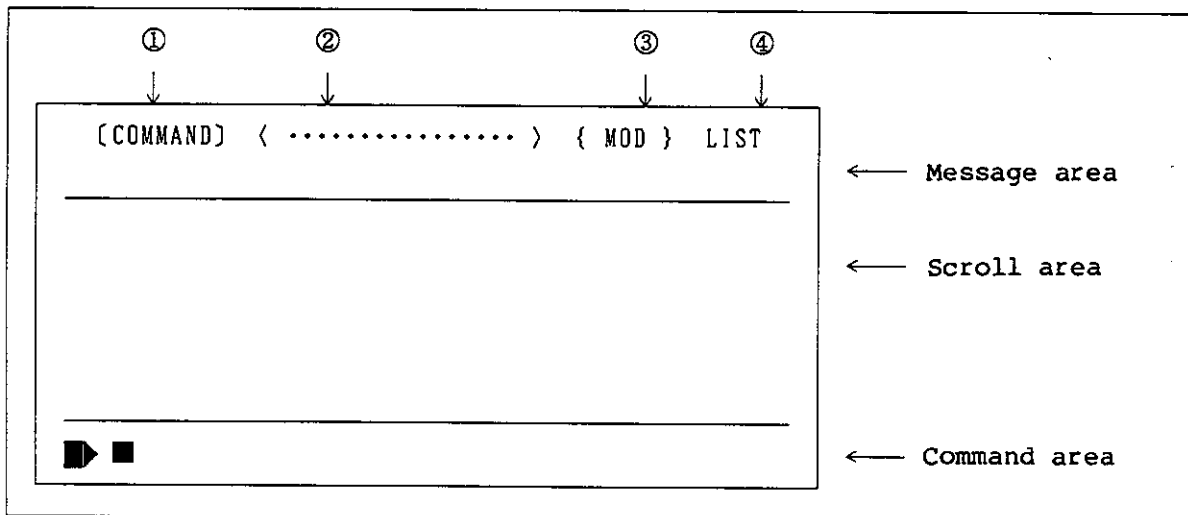


Figure 4 - 1 CRT Display During Program Mode

① Display current input mode

[COMMAND] When cursor is on input line

[EDITOR] When cursor is in scroll area

② Display file name which can currently be edited

<.....> ... New file being generated, or no file loaded

<file-name> ... Name of file currently loaded

NETWORK ANALYZER
PROGRAMMING MANUAL

4.2 Activation of Program Mode

③ Display editor mode status

{OK} File correctly loaded
{NG} File not correctly loaded
{NEW} New file being generated
{MOD}..... Editing existing file
{APN}..... Adding to existing file
{?} Command mode

④ When a function key is pressed, that function is displayed.

Input mode may be either command or editor. The initial mode set is command mode where all input data (maximum of 45 characters) is typed in on the input line. Direct input to the scroll area is not possible at this stage.

(2) Commands and Programs

When a statement following a line number is keyed in, that line becomes a program line. If a statement is typed in and executed without specifying a line number, the line is called a command.

Example:

▶ 10 PRINT "BASIC" Program
▶ LIST 10 100 Command

(3) Input and Execution

To input a program line, type in a line number followed by a valid statement, and then press the RETURN or ENTER key.

That line is then stored in memory as part of a program. That line is not executed until the program itself is executed.

When executing a new program, always remove the old program by typing in SCRATCH from the keyboard.

Example:

▶ SCRATCH

NETWORK ANALYZER
PROGRAMMING MANUAL

4.2 Activation of Program Mode

The SCRATCH statement is used to initialize previous input programs and variables.

SCRATCH Initialization of programs and variables

SCRATCH 1 Initialization of variables

SCRATCH 2 Initialization of programs

4.3 Editor Mode Activation

Program input in command mode requires input of line numbers. And since program lines are cleared once the end is reached, it is very difficult to know the current position of the program, or to collate a program which has already been entered. The editor mode is used to overcome this problem.

● Editor Mode

Editor mode is activated by typing in EDIT and pressing the RETURN key. As a result, the display shown below appears on the CRT screen.

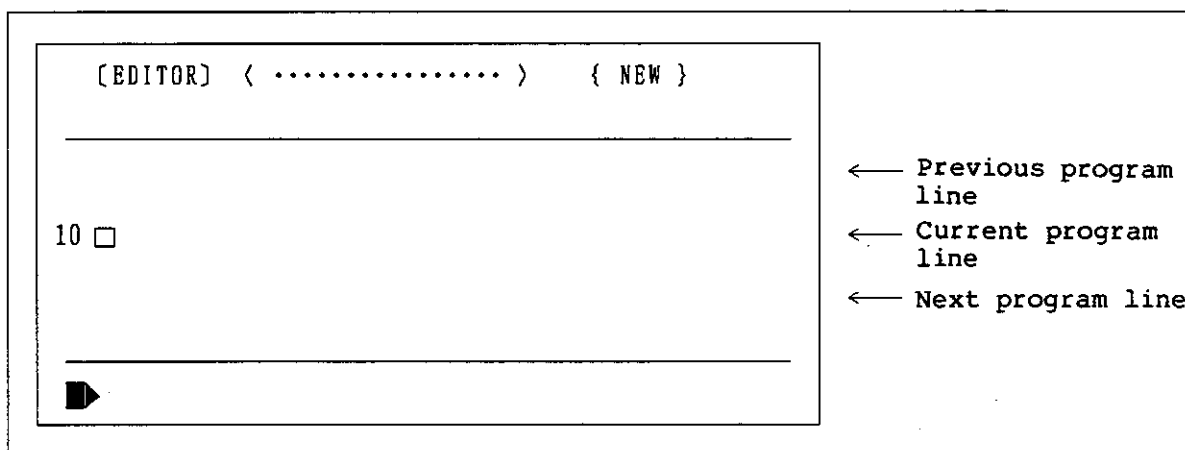


Figure 4 - 2 CRT Display During Editor Mode

Line number are displayed automatically in editor mode. Two parameters can be specified in the EDIT command. There are the initial line number, and the line increment. For example, the command

```
EDIT 100
```

Specifies that line 100 of the file current in the editor area is to be displayed in the center of the CRT screen, and the cursor is set at the end of that line.

If no parameters are specified, the following default values are used.

```
Initial line number : 10  
Line increment      : 10
```

But where a previous program is currently being edited, the line increment parameter value is disregarded.

NETWORK ANALYZER
PROGRAMMING MANUAL

4.4 Program Editor Keys

4.4 Program Editor Keys

An optional keyboard (TR45103) is used to input programs. This keyboard is connected to the network analyzer which is then set to program mode. Note that apart from some panelkeys and softkeys, none of the network analyzer functions can be used when in program mode.

CAUTION

Since disconnecting the external keyboard connector during operation results in generation of an error, always switch the power off before connecting or disconnecting this connector.

The keyboard conforms with the JIS layout. Together with shift positions (with the SHIFT key depressed), standard ASCII characters including alphanumeric characters and special signs can be typed in.

① Special Keys

SHIFT

Used to key in characters in the shift position of each key. And when keying in alphabetic characters, the SHIFT key is used to key in upper case characters. If the CAPS LOCK key is locked, lower case characters are keyed in.

CTRL








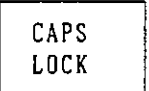



Used in combination with other keys for input of special codes.

Table 4 - 1 CTRL Key Operation

Key input	Operation
CTRL + C	Suspend program or command execution
CTRL + D	Reset if editor fails
CTRL + G	Activate buzzer
CTRL + H	Delete character to left of cursor (same action as BACK SP key)
CTRL + I	Same as pressing TAB key
CTRL + J	LINE FEED Move cursor to beginning of line
CTRL + M	Terminate program input (same as RETURN key)
CTRL + Q	Same as pressing NO SCROLL key once
CTRL + S	Same as pressing NO SCROLL key twice

NETWORK ANALYZER
PROGRAMMING MANUAL

4.4 Program Editor Keys

	Press to terminate input of one line. In editor mode, the cursor moves to the beginning of the next line. In command mode, the input line is cleared, and the cursor moves to the beginning of the line.
	No function
	Move cursor one character to the right.
	Move cursor one character to the left.
	Move cursor one line upwards. If the cursor is already at the top line, the entire program is scrolled down by half a page, and the cursor moves to the center of the CRT screen.
	Move cursor one line downwards. If the cursor is already at the bottom line, the entire program is scrolled up by half a page. There is no action when in command mode.
	Delete the character at the cursor position.
	When this key is locked by pressing, all subsequent input characters are keyed in as upper case characters. The key is unlocked by pressing a second time.
	Used to cancel editor mode, and to switch to command mode.
	Input of two spaces.
	Delete character to the left of the cursor.

② Function Keys

First check that the function key name plate is 09. This name plate is divided into two upper rows with the function name printed on each key. Normally, only the lower row of functions is used. To use the upper row functions, the keys have to be pressed together with the SHIFT key.

NETWORK ANALYZER
PROGRAMMING MANUAL

4.4 Program Editor Keys

Table 4 - 2 Function Operations

Function name	Command	Edit	Measuring mode
↓	x	o	x
⇓	x	o	x
↑	x	o	x
⇑	x	o	x
LIST	o	o	x
DEL LN	o	o	x
INS LN	x	o	x
CLR LN	o	o	x
F1 (LOAD ")	o	●	●
F2 (SAVE ")	o	●	●
F3 (SCRATCH)	o	●	●
F4	x	●	●
F5	x	●	●
F6	x	●	●
CAT	o	x	x
EDIT	o	x	x
CHKDSK	o	x	x
CHG MODE	o	x	o
NEXT	o	o	x
PREV	o	o	x
CLS	o	x	x
PAUSE	o	x	x
CONT	o	x	x
STOP	o	x	o
STEP	o	x	x
RUN	o	x	o

- : Partial functioning
- o : Function activated
- x : No function

● Description of Functions



Scroll up program by one line without changing cursor position.



Scroll up by half a page and move cursor to center line.



Scroll down program by one line without changing cursor position.



Scroll down by half a page and move cursor to center line.

NETWORK ANALYZER
PROGRAMMING MANUAL

4.4 Program Editor Keys

LIST	Commence display of program from beginning when in command mode, or redisplay current screen when in editor mode.
DELIN	Delete cursor line and line number.
INSLN	Open space equivalent to one character on the line where cursor is located, and display a suitable minimum line number in that space. If insertion between lines is not possible, a message is displayed to recommend that no insertion be attempted.
CLRLN	Clear current cursor line without erasing line number.
F1 to F6	See this manual part 1 (4.6 "Function Keys"). (Note that F1 thru F3 contain commands.)
CAT	Display CAT on command line.
EDIT	Display EDIT on command line.
ACHKDSK	Display floppy disk information.
CHG MODE	Switch menu screens for command and measuring modes.
PREV	Restore previous command executed in command mode.
NEXT	Reverse the result of executing PREV in command mode.
CLS	Clear editor screen, and set display start line at beginning.
PAUSE CONT STOP STEP and RUN	correspond to BASIC commands.

NETWORK ANALYZER
PROGRAMMING MANUAL

4.4 Program Editor Keys

CAUTION

1. Use of the INS LN and DEL IN function keys may on odd occasions result in cursor or line number malfunction. If this happens, press LIST (redisplay screen) once or twice to correct the display and resume editing.
2. The editor screen may deteriorate when using the CURSOR command in editor mode. In this case, press Ctrl-D (reset editor) to return to normal editor display.
3. When the last line of a program is specified at the EDIT line number, the same line may appear twice on the screen. In this case, press LIST to return to normal.

4.5 Program Editing

① Input of Program Lines

To input a program line, type the program line after the line number, and then press the RETURN key.

In editor mode, line numbers are given automatically, but input or changing of line numbers is not possible.

② Insertion of Characters

To insert a character in a line which has already been programmed or which is about to be programmed, a single character can be inserted at the position of the cursor.

When a character is keyed in to be inserted at the cursor position, all characters from that position up to the end of the line after shifted to the right by one character.

Although the screen display is changed, the actual program will remain unchanged of the RETURN key is not pressed.

③ Deletion of Characters

Characters can be deleted during programming by pressing the DEL or BS key. The character at the cursor position is deleted when the DEL key is pressed, and all characters to the right of that position are shifted to the left by one character.

When the BS key is pressed, the character to the left of the cursor is deleted, followed by left justification.

④ Insertion of Lines

Use "INS_LN" to insert a new line. For example, to insert a line between lines 130 and 140 in the following program, first move the cursor to the beginning of line 140. When "INS_LN" is pressed, line 131 is displayed waiting for the input data. If more than one line is inserted at this stage and RETURN is pressed, "Illegal insert line" is displayed. Therefore, first exist from editor mode, execute the REN command, and repeat the above procedure.

```
130 PRINT "KEY NUMBER ?"  
140 OUTPUT 31: "CH1"  
  
130 PRINT "KEY NUMBER ?"  
131 _  
140 OUTPUT 31: "CH1"
```

⑤ Clearing and Deletion of Lines

Lines may be removed by clearing (CLR_LN) or deleting (DEL_LN). Whereas "clearing" refers to removal of a program line without removing the line number, "deleting" refers to removal of the program line plus the line number.

(CLR_LN)

```
130 PRINT "KEY NUMBER ?"  
      (Removed data)
```

(DEL_LN)

```
140 PRINT "KEY NUMBER ?"  
      (Removed data)
```

And when in COMMAND mode, the DEL command is used to delete data. Two specifiers can be specified in the DEL command. The first number specifies the line number at the beginning of the block to be deleted, and the second number specifies the line number at the end of the block.

```
DEL 100          Delete line 100.  
DEL 100, 200    Delete from line 100 to line 200.
```

⑥ Rearranging Program Numbers

If editing involves the deletion and insertion of many lines, the line numbers can be rearranged to make the program easier to read. This feature is also useful where many additional lines are inserted. Line numbers are rearranged by using the REN command. The first line number and the line increment can be specified.

For example, specifying

```
REN 50 100 5
```

results in the lines of the entire program (where the first line number is 50) currently stored in memory being renumbered from line 100 in line increments of 5. The default line increment value is 10.

⑦ Generation of Program List

Execute the LIST statement to display the entire program (or part of it) on the CRT screen. The range of lines to be shown can be specified in the LIST statement.

NETWORK ANALYZER
PROGRAMMING MANUAL

4.5 Program Editing

LIST 100 Display line 100 only.
LIST 100, 200 Display from line 100 to line 200.
LIST Display entire program.

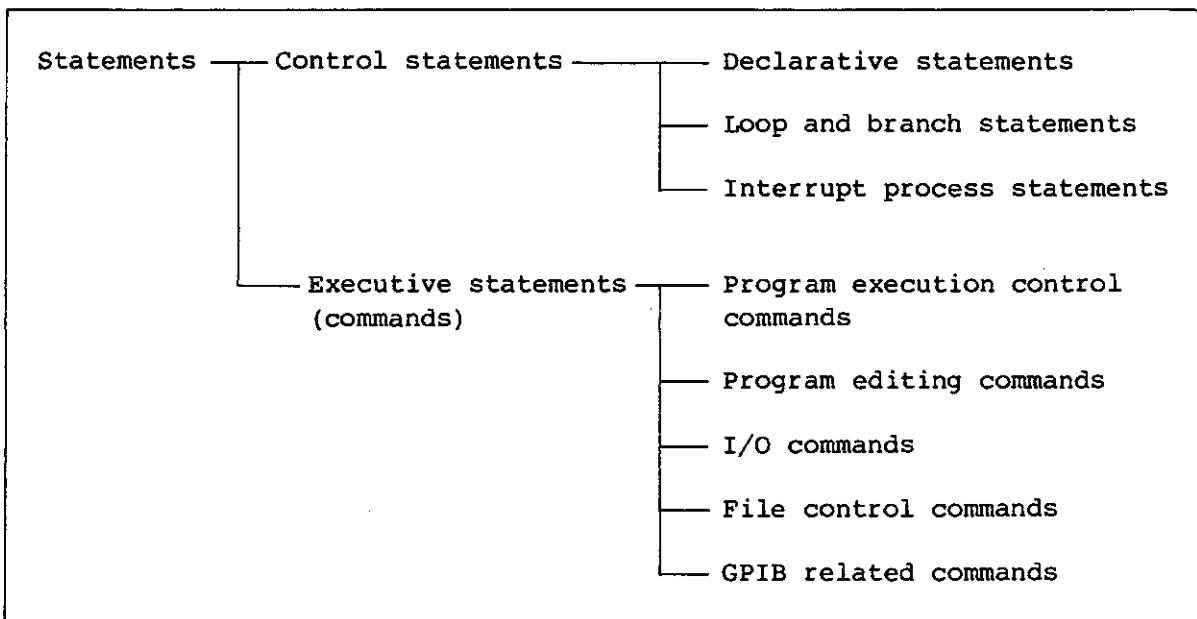
LISTN 100, 10 Display 10 lines from line 100.

4.6 Programming Rules

4.6.1 Program Architecture

BASIC programs are collections of various types of statements.

Statements are divided into two types - control statements and executive statements (commands).



Each statement consists of a key word and expression, and this configuration is determined by grammatical syntax rules.

BASIC words whose meaning and applications have been decided in advance are called key words. Therefore, the same names as key word names cannot be used for any other purposes.

NETWORK ANALYZER
PROGRAMMING MANUAL

4.6 Programming Rules

A list of key words is given in the following table.

Table 4 - 3 List of Key Words

AND	APPEND	AS	ASCII	BAND	BASIC
BINARY	BNOT	BOR	BREAK	BUZZER	BXOR
CASE	CAT	CHKDSK	CLEAR	CLOSE	CLS
CMD	CONT	CONTINUE	CONTROL	COPY	COPYFILES
COUNT	CSR	CURSOR	DATA	DEL	DELIMITER
DIM	DISABLE	DSTAT	DUMP	ELSE	ENABLE
END	ENT	ENTER	ENTERF	ERROR	FOR
FORMAT	GLIST	GLISTN	GOSUB	GOTO	GPRINT
IF	INIT	INITIALIZE	INP	INPUT	INTEGER
INTERFACE	INTR	ISRQ	KEY	LABEL	LIST
LISTEN	LISTN	LLIST	LLISTN	LOCAL	LOCKOUT
LPRINT	LOAD	MERGE	NEXT	NEWVERSION	NOT
OFF	ON	OPEN	OR	OUTPUT	OUT
OUTPUTF	PAUSE	PRINT	PRINTER	PRF	PRINTF
READ	RESTORE	PURGE	RENAME	REM	REMOTE
REN	REQUEST	RETURN	RUN	SAVE	SCRATCH
SELECT	SEND	SPRINTF	SRQ	STEP	STOP
SYSTEM	TALK	TEXT	THEN	TIME	TO
TRIGGER	UNL	UNT	UNTIL	USE	USING
XOR					

Shorten name is used for entering a key word. Shorten names are provided for the frequently used and long key words. Shorten name can be used as a key word. On the display, shorten name is used when control register of 3 is set to 1 by CONTROL command. To display in full name, set the control register of 3 to 0.

(Correspondence of full name and shorten name)

Full name	Shorten name
CURSOR	CSR
ENTER	ENT
INITIALIZE	INIT
INPUT	INP
OUTPUT	OUT
PRINTF	PRF
USING	USE
PRINT	?

● Expressions

Expressions consist of objects and operators, and can be placed anywhere within the syntax where an expression can be specified.

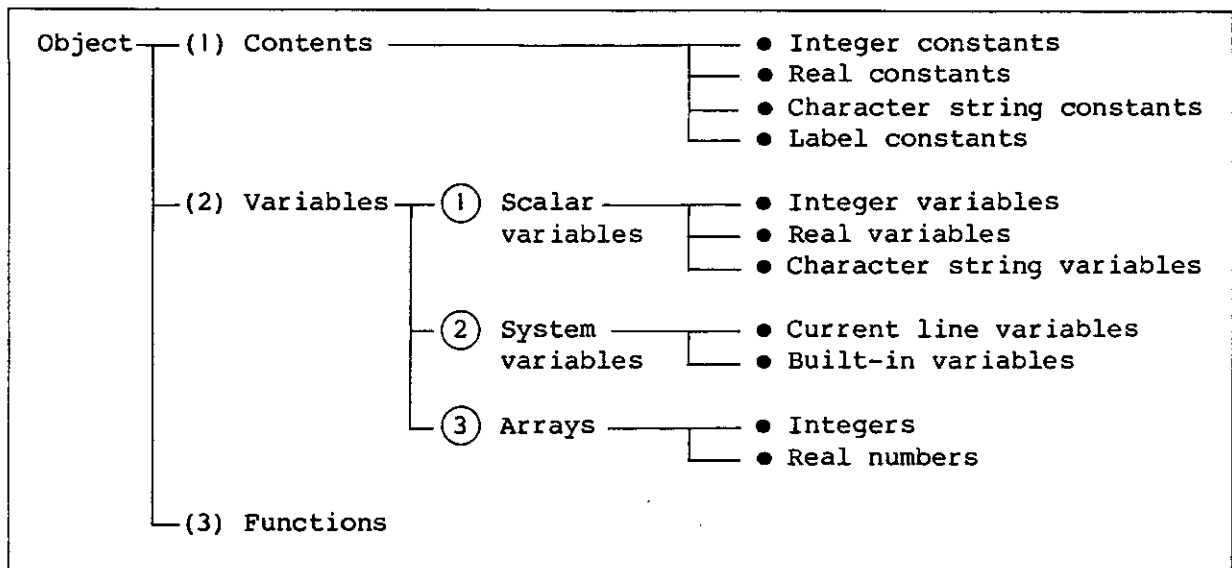
(To maintain compatibility with conventional BASIC, however, substitute expressions cannot be written in IF statement conditional expression since "=" is interpreted as a code.)

Expressions include, arithmetic expressions, character string expressions, logical expressions, and label which differ according to the data format in which the final calculated value is obtained. Arithmetic expressions consist of integer and real numbers. Logical expressions are determined by syntax, irrespective of whether the expression contains logical operators, the final value being evaluated as a logical value. That is, 0 is false, and anything else is true.

4.6.2 Objects

Elements subject to BASIC processing are called objects. These objects contains a constant, variable, and function.

Each data type is as below.



(1) Constants

● Real Constants

Numerical values with no decimal point are regarded as integer numbers in program. Since these can be expressed internally in 4 bytes, numbers can be expressed from -2,147,483,648 to 2,147,483,647.

● Real Numbers

Numerical values containing a decimal point, or expressed as an exponential number like 1E+20 are regarded as real numbers. And since these can be expressed internally by using 8 bytes (IEEE), numbers from about -1E+308 to 1E+308 can be represented with an accuracy of 15 digits.

● Character String Constants

Character strings are expressed by being enclosed between double quotation marks ("").

Character strings can be specified as a null character string (" "), or as strings containing up to 128 characters. The component character unit is 8 bits which allows a maximum of 256 different character units to be expressed. The ASCII character code is used, and characters 128 thru 255 are registered with special symbols.

Reference:

To express (by program) codes not assigned to the keyboards, and to input data by INPUT statement, (\) is used in a method called \014 (form field). Likewise, to include double quotation marks (") inside a character string, this may be written as (\).

The following escape sequence is provided to express ASCII control characters.

	Octal	Decimal	
\b	010	8	Back space
\t	011	9	Vertical tabulation
\n	012	10	Line feed (New line)
\v	013	11	Vertical tabulation
\f	014	12	Form feed (Clearing screen)
r	015	13	Carriage return

● Label Constants

Label constants are used instead of statement numbers, and are declared by appending an asterisk (*) at the start of a program.

Although the characters which can be used as the same as those for variables, substitution is not possible because they are not variables. And places where labels can be written are restricted by syntax. The places are the part in the later section that "Label line number" or "Branch destination" is written.

(2) Variables

Variable names consist of up to 20 alphanumeric characters starting with an alphabetic character.

Table 4 - 4 Alphanumeric Characters

1, 2, 3, 4, 5, 6, 7, 8, 9, 0,
a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t,
u, v, w, x, y, z
A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T,
U, V, W, X, Y, Z
-

Variable names become character string variables if \$ appended to the end. And if ... is appended to the end of a variable name instead of a \$, that variable becomes an array type variable. If a variable is not specifically declared by INTEGER statement, it becomes a real number type of variable.

Examples of variable types:

value, v123	Real variables
string\$, s123\$	Character string variables
array(3)	Array type real variable
INTEGER code	Integer variable
INTEGER week(7)	Array type integer variable

① Scalar Variables

- Integer variables
- Real variables
- Character string variables

Numeric variables are allocated the value 0 unless specifically initialized. Therefore, variables to initialized to a specific value must have a specific value substituted in them in the program.

The size of values which can be stored in each data type are the same as for constants.

There are no array character string variables. Like character string constants, character strings include a length attribute. The DIM statement is used to declare length.

```
DIM string$[100]
```

If collating without a declaration, the default character string length is 18 characters.

By using a sub string operator ([]), certain parts of the character string can be handled (see (6) sub string operator in item 4.6.3).

```
string$ = "ADVANTEST CORPORATION"  
PRINT string$[1,14];"."
```

Result:

```
ADVANTEST CORP.
```

② System Variables

● Current Line Variables @

Storage of the program line number currently be executed. Values cannot be substituted.

LIST@ : Display of the line currently being executed.

● Built-in Variables

Built-in variables are registered automatically when BASIC is started up. These are initialized by fixed values, and can be substituted by specific values. To return to the original value, either explicitly substitute that value, or initialize by using SCRATCH 2, SCRATCH.

```
PI : 3.14152 .....  
EXP : 2.71828 .....
```

③ Array

Use the DIM or INTEGER statement to declare an array.

● Numeric Array

If collating without a declaration, the array size (that is, number of elements) is 10. The result is the same as when declaring as below.

```
DIM array(10)
INTEGER array(10)
```

```
Real number array DIM real(20)
Integer number array INTEGER int(30, 40)
```

(3) Functions

All functions are built-in functions, and are divided into integer, real number, and character string types in terms of the return value. And since function calls can be described in operational expressions, functions can be handled in the same way as variables.

```
string$ = "ADVANTEST"
PRINT string$
A = NUM("A")
a = NUM("a")
FOR idx = 0 to LEN(string$)
    b = NUM(string$[idx:1]) - A + a
    string$[idx:1] = CHR$(b)
NEXT idx
PRINT string$
```

Result:

```
ADVANTEST
advantest
```

Built-in functions

NUM(character string expression)

The ASCII code of the leading character of the character string expression is returned.

NUM("A") → 65

CHR\$(arithmetic expression)

The character string expression of the single ASCII character corresponding to the arithmetic expression value is returned.

CHR\$(65) → "A"

NETWORK ANALYZER
PROGRAMMING MANUAL

4.6 Programming Rules

LEN(character string expression)

Length of character string expression is returned.

LEN("ADVANTEST") → 9

POS(character string expression 1, character string expression 2)

The start position of a certain position in character string expression 2 is returned from character string expression 1.

POS("ADVANTEST", "AN") → 4

SIN(arithmetic expression)

COS(arithmetic expression)

TAN(arithmetic expression)

ATN(arithmetic expression)

LOG(arithmetic expression)

SQR(arithmetic expression)

In addition to those listed below, a wide range of this instrument built-in functions capable of handling measured values is available. See the list of built-in functions in section 5.1 "Built-in Functions".

--- built-in function ---

Frequency -----> Point No.

POINT1(F, M)
POINT2(F, M)
DPOINT(F₀, F₁, M)

Point No. -----> Frequency

FREQ(P, M)
DFREQ(P₀, P₁, M)

Point No. -----> Response Value

VALUE(P, M)
DVALUE(P₀, P₁, M)

Frequency -----> Response Value

CVALUE(F, M)
DCVALUE(F₀, F₁, M)

Searching Maximum

MAX(P₀, P₁, M)
FMAX(P₀, P₁, M)
PMAX(P₀, P₁, M)

NETWORK ANALYZER
PROGRAMMING MANUAL

4.6 Programming Rules

Searching Minimum

MIN(P₀, P₁, M)
FMIN(P₀, P₁, M)
PMIN(P₀, P₁, M)

Calculate Bandwidth

BND(P, X, M)
BNDL(P, X, M)
BNDH(P, X, M)
CBND(F, X, M)
CBNDL(F, X, M)
CBNDH(F, X, M)

Differential coefficient

DIFFX(Δ X, Δ Y, M)
DIFFY(Δ X, Δ Y, M)

Finding Ripple out

RPL1(P₀, P₁, Δ X, Δ Y, M)
RPL2(P₀, P₁, Δ X, Δ Y, M)
RPL3(P₀, P₁, Δ X, Δ Y, M)
RPLF(P₀, P₁, Δ X, Δ Y, M)
RPLR(P₀, P₁, Δ X, Δ Y, M)

Megalo and micro detection

RPLH(P₀, P₁, Δ X, Δ Y, M)
FRPLH(P₀, P₁, Δ X, Δ Y, M)
PRPLH(P₀, P₁, Δ X, Δ Y, M)
RPLL(P₀, P₁, Δ X, Δ Y, M)
FRPLL(P₀, P₁, Δ X, Δ Y, M)
PRPLL(P₀, P₁, Δ X, Δ Y, M)
NRPLH(P₀, P₁, Δ X, Δ Y, M)
NRPLL(P₀, P₁, Δ X, Δ Y, M)
PRPLHN(N, M)
PRPLIN(N, M)
FRPLHN(N, M)
FRPLLN(N, M)
VRPLHN(N, M)
VRPLLN(N, M)

Testing limit

LMTUL1(X, Up, L₀, M)
LMTUL2(P, Up, L₀, M)
LMTMD1(X, Up, L₀, M)
LMTMD2(P, Up, L₀, M)

Phase 0 detection

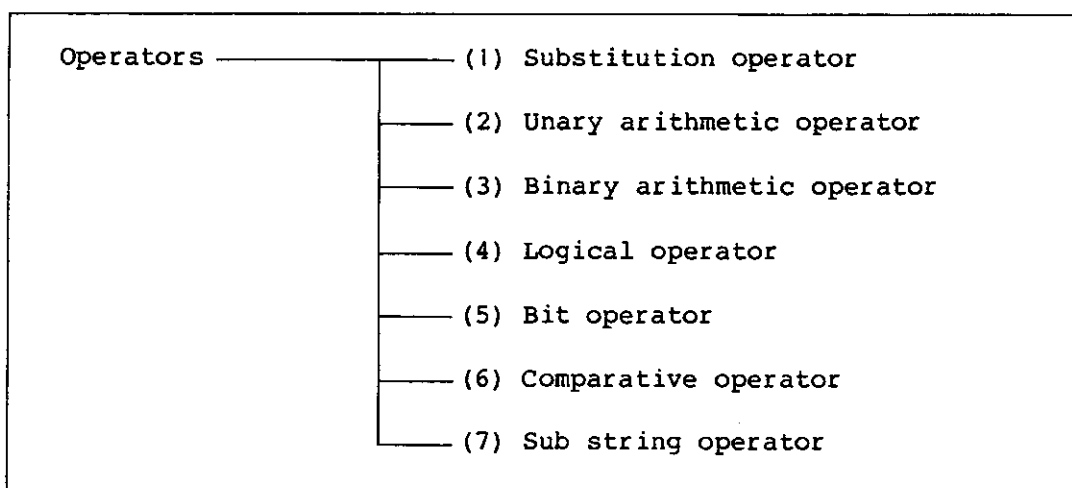
ZEROPHS(P₀, P₁, M)

Direct search

DIRECT(P₀, P₁, X, M)
CDIRECT(F₀, F₁, X, M)
DDIRECT(P₀, P₁, X, M)
CDDIRECT(F₀, F₁, X, M)

4.6.3 Operators

Objects are manipulated by operators, and objects and operators are combined in expressions.



(1) Substitution Operators

The conventional BASIC keyword "LET" has not been included. The substitution operator contains its own value to become a single expression.

```
PRINT a = 1           —> 1  
PRINT a$="ADVANTEST" —> "ADVANTEST"  
PRINT (a=1)+a        —> 2
```

The substitution operator contains the following elements.

= Normal substitution

In character string substitutions, the valid characters on the right hand side are transferred.

Example :

```
INTEGER string${20}
PRINT LEN(string$ = "12345")
```

Result : 5

Substitution after conversion to data format on left hand side of =.

Example :

```
string$ = 123.456 ----> "123.456"
numeric = "123" ----> 123
integer = 123.456 ----> 123
```

+= a += 10 <==> a = a + 10

-= a -= 10 <==> a = a - 10

*= a *= 10 <==> a = a * 10

/= a /= 10 <==> a = a / 10

%= a %= 10 <==> a = a % 10

=< Substitute after left justification of character string.

=> Substitute after right justification of character string.

(2) Unary Arithmetic Operators

- : Minus sign

+ : Plus sign

++ : Pre-/post-increment

Pre- a = 1 : b = ++a

Substitute in b after adding 1 to a.

Post- a = 1 : b = a++

Add 1 to a after substitute in b.

-- : Pre-/post-decrement

Pre- a = 1 : b = --a

Substitute in b after subtracting 1 from a.

Post- a = 1 : b = a--

Subtract 1 from a after substituting in b.

Example:

```
a = 10 : PRINT a++ : PRINT a : PRINT --a : PRINT --a : PRINT a
```

Result : 10.

11.

10.

9.

9.

(3) Binary Arithmetic Operators

+ : Addition
- : Subtraction
* : Multiplication
/ : Division
% : Modulo (remainder)
^ : Involution
& : Character string concatenation

(4) Logical Operators

NOT
AND
OR
XOR

(5) Bit Operators

They execute the 16-bit calculation. Only the integer type equations can be set. If a real type equation is set, an error occurs.

BNOT
BAND
BOR
BXOR

(6) Comparative Operators

The following comparative operators are used. 1 is taken if result is true, and 0 if false. When a comparative operation is executed in BASIC syntax, and the final result is 0, this is taken as false. All other results are taken as true.

= : Equal (or ==)
< >: Not equal (or !=)
<
>
<=
>=

Since this comparative operator must always execute a logical operation in IF statement conditions, the "=" operator is regarded as a unconditional comparative operator. Therefore, substitution expressions cannot be included in IF statement condition expression.

NETWORK ANALYZER
PROGRAMMING MANUAL

4.6 Programming Rules

To execute comparison operations apart from using an IF condition expression, "==" is used for equal operation purposes to make a distinction from "=" used in substitution operators.

```
a = (b$ == "COMPUTER")
```

If the character variable b\$ is "COMPUTER", variable a is 1.

(7) Sub String Operator

Character string expression parts can be specified as a character string.

Character string expression [arithmetic expression 1, arithmetic expression 2]

The section of a character string expression where arithmetic expression 1 has advanced from the beginning of the string expression by the indicated value up to the value where arithmetic expression 2 is indicated is the sub string.

```
"ADVANTEST"[1,5] → "ADVAN"
```

Character string expression[arithmetic expression 1, arithmetic expression 2]

The number of characters in a character string expression where arithmetic expression 1 has advanced from the beginning of the string expression by the indicated value up to the value where arithmetic expression is indicated is the sub string.

```
"ADVANTEST"[6;4] → "TEST"
```

(8) List of character code

- Character code

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0																
1																
2																
3																
4																
5																
6																
7																
8																
9																
a																
b																
c																
d																
e																
f																

- Example: PRINT CHR\$ (0 x 41)

- Result: A

5. COMMAND AND STATEMENT SYNTAX AND COMMENTARY

5.1 Outline

The command and statement syntax used in the network analyzer is described here in combined diagrammatical/textual format to make it easier to understand.

CAUTION

How to read **Syntax** of each command or statement.

(1) Diagrammatical Representation

The syntax is divided into component elements linked up by straight lines.

Statements always proceed in the direction indicated by arrow. If branching occurs, the statement proceeds along one of those branches. And where a loop is formed, that loop may be passed any number of times.

(2) Textual Representation

The following symbols are used in textual representation.

[] : Sections enclosed by this symbol may be omitted.

< > : Sections enclosed by this symbol may be omitted.

{ } : Sections enclosed by this symbol may be used any number of times.

| : This symbol denotes "or".

(Example : <A>| ... Use either <A> or .)

(3) Terminology used in these diagrammatical and textual representations is described below.

- Numerical value representation
... Numerical constant, numerical variable, or numerical expression
- Character string representation
... Character string constant, character string variable, character string function, or expression consisting of substrings.
- Device address
... Address of device connected to GPIB

NETWORK ANALYZER
PROGRAMMING MANUAL

5.2 List of Commands and Statements

5.2 List of Commands and Statements

(1) Commands

CAT : Outputs file name on CRT screen
CHKDSK : Displays disk status
CONT : Resumes program execution
CONTROL : Sets the various BASIC control variables
COPY : Copies file

DEL : Deletes specified line number
DUMP : Indication in the memory and file
EDIT : Starts editor mode
FRE : Indication of the basic program buffer remain
GLIST : Outputs program list to GPIB

GLISTN : Outputs program list to GPIB
INITIALIZE : Initializes floppy disk
LIST : Displays program list on CRT screen
LISTN : Displays program list on CRT screen
LLIST : Outputs program list to serial port

LLISTN : Outputs program list to serial port
LOAD : Loads BASIC program from floppy disk
MERGE : Loads and merge program with another program
PRINTER : Sets printer GPIB address
PURGE : Deletes file from disk

REN : Renumbers line numbers
RENAME : Changes file name
RUN : Executes a program
SAVE : Saves BASIC program to floppy disk
SCRATCH : Deletes previously loaded program

STEP : Executes one line of program

(2) Statements

BREAK : Exits FOR-NEXT block
BUZZER : Buzzer
CASE : Defines conditions
CLS : Clear screen
CONTINUE : Branches to loop of next step value from
FOR-NEXT loop

CURSOR : Cursor position control

NETWORK ANALYZER
PROGRAMMING MANUAL

5.2 List of Commands and Statements

DATA	:	Defines numerical values and character strings to be read in the READ statement
DIM	:	Declares array variables
DISABLE INTR	:	Disable interrupt branching
ENABLE INTR	:	Enable interrupt branching
ERRM\$:	Returns error message
ERRN	:	Returns error code
FOR-TO-STEP	:	Executes loop processing
GOSUB	:	Branches to subroutine
GOTO	:	Branches to specific line
GPRINT	:	Outputs numerical values and character strings to GPIB
IF THEN	:	Conditional branching
INPUT	:	Input from keyboard
INTEGER	:	Defines variable as integer number
LPRINT	:	Outputs numerical values and character strings to serial port
NEXT	:	Executes loop processing
OFF ISRQ	:	Releases interrupt branching generated by ISRQ
OFF KEY	:	Releases interrupt branching generated by KEY input
OFF SRQ	:	Releases interrupt branching generated by SRQ
ON ERROR	:	Defines interrupt branching to be executed if BASIC error is detected
ON ISRQ	:	Defines interrupt branching by the network analyzer internal source
ON KEY	:	Defines interrupt branching by KEY input
ON SRQ	:	Defines interrupt branching by GPIB external SRQ signal
PAUSE	:	Halts program execution temporarily
PRINT[USING]	:	Displays (output) of numerical values and character strings
PRINTER	:	Sets GPIB address for printer
PRINTF	:	Displays (output) of numerical values and character strings
READ	:	Replaces constants in the DATA statements with variables
REM	:	Comment
RESTORE	:	Defines DATA lines to be read in the next DATA statement
RETURN	:	Returns from subroutine
SELECT	:	Branches as conditioned by values of the equation
SPRINTF	:	Replaces character strings with results of PRINTF format

NETWORK ANALYZER
PROGRAMMING MANUAL

5.2 List of Commands and Statements

(3) GPIB control statements

CLEAR : Clear equipment
DELIMITER : Specifies block delimiter
ENTER : Input from GPIB
INTERFACE CLEAR : Clear GPIB interface
LOCAL : Releases remote control

LOCAL LOCKOUT : Local lockout
OUTPUT : Outputs to GPIB
REMOTE : Remote control
REQUEST : Sets status byte
SEND-DATA-CMD-TALK-LISTEN-UNT-UNL : Outputs of commands and data to GPIB

TRIGGER : Outputs of group execute trigger

(4) File control statements

CLOSE : Closes files for file descriptor
COPYFILES : Copies files to another floppy disk
ENTER [USING] : Reads data in files
OFF END : Releases processing specified by the ON END statement
ON END : Defines end-of-file processing

OPEN : Opens files for file descriptor
OUTPUT [USING] : Writes (output) data into the file

5.3 BASIC Command Syntax

This section explains the following commands in numeric order.

1. CAT
2. CHKDSK
3. CONT
4. CONTROL
5. COPY
6. DEL
7. DUMP
8. EDIT
9. FRE
10. GLIST
11. GLISTN
12. INITIALIZE
13. LIST
14. LISTN
15. LLIST
16. LLISTN
17. LOAD
18. MERGE
19. PRINTER
20. PURGE
21. REN
22. RENAME
23. RUN
24. SAVE
25. SCRATCH
26. STEP

NETWORK ANALYZER
PROGRAMMING MANUAL

5.3 BASIC Command Syntax

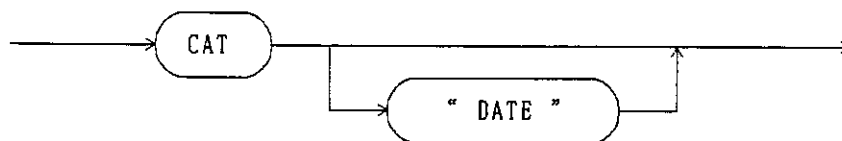
1. CAT

Outline

Displays of file stored on floppy disk.

Syntax

(1)-1



(1)-2

CAT ["DATE"]

Commentary

- Display of contents of file stored on disk.
When CAT is used, the registration number, file name, number of sectors used, number of characters, and file attributes are displayed in that order. And by using CAT "DATE", the registration number and file name are followed by the date and time when the file was generated.

2. CHKDSK

Outline

Displays status of disk in disk drive.

Syntax

(1)-1

→ (CHKDSK) →

(1)-2

CHKDSK

Commentary

- Display status of disk in disk drive. This information includes:

DISKNAME ... Disk name applied during initialization

FILES Number of files

SECTOR Number of sectors used

DATE Date and time of initialization

Where:

FILES are up to 200.

SECTORS are up to 1400.

SECTOR is a unit of information stored on a disk.

1 SECTOR is equal to 512 bytes.

Example

The following display appears when CHKDSK is executed immediately after initialization.

<DISK-ID>

[DATE : 1988.01.15 (Fri) 13:05]

[FILE : 0 / 200]

[SECTOR : 0 / 1400]

[DISKNANE : ADVANTEST_NA]

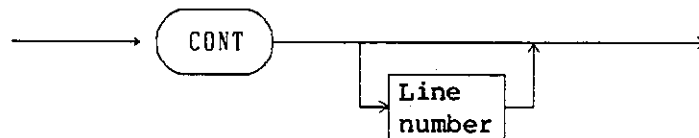
3. CONT

Outline

Resumes execution of BASIC program.

Syntax

(1)-1



(1)-2

CONT [Line number]

Commentary

- Execution of BASIC program is resumed from specified line.
- Variables are not initialized by CONT command.

Example

CONT 200

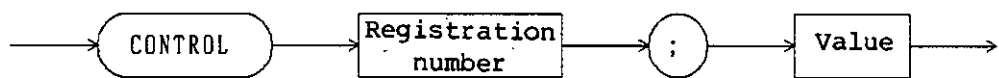
4. CONTROL

Outline

Sets various values related to BASIC control.

Syntax

(1)-1



(1)-2

CONTROL <Registration number> ; <Value>

Commentary

Specify control elements to be set by registration number. Values following the semicolon are actual settings.

Registration number

(Registor 1)

Serial I/O port initialization

Specifies by the summation of the following values.

Value : Band rate

- 0 : 1200 baud
- 1 : 2400 baud
- 2 : 4800 baud
- 3 : 9600 baud

Character length

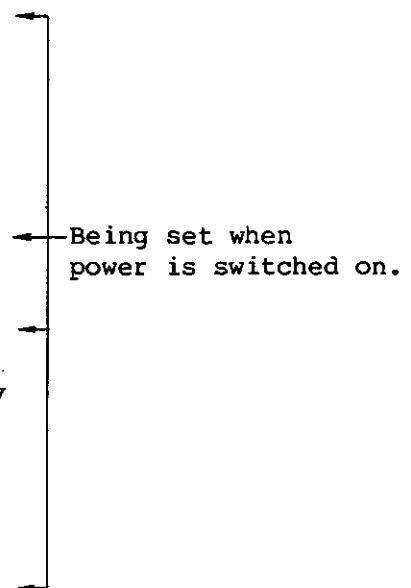
- 0 ; 5 bits
- 4 ; 6 bits
- 8 ; 7 bits
- 12 ; 8 bits

Parity

- 0 ; No parity
- 16 ; Odd parity
- 48 ; Even parity

Number of stop bits

- 0 ; None
- 64 ; 1 bit
- 128 ; 1 1/2 bit
- 192 ; 2 bits



NETWORK ANALYZER
PROGRAMMING MANUAL

5.3 BASIC Command Syntax

(Register 2)

The printing position from the left hand margin is specified by the number of spaces with LLIST/GLIST.

(Register 3)

Selects whether the BASIC program is indicated in shorten name or conventional full name.

When 1 is set, BASIC program is indicated in shorten name. When 0 is set, full name indication is selected.

(Register 5)

Register 5 is used to change the environment to that for maintenance.

When register 5 is set to 1, POKE command is effective. If register 5 is set to 0, POKE command is invalid.

(Register 6)

Specifies the termination of INPUT statement (1 or 0).

When the ENTER key or function key is pressed for 1, the statement is terminated. When the ENTER key is pressed as usual for 0, it is terminated.

Example

Registration number 1

To set baud rate to 9600, character length to 8 bits, even parity, and 2 stop bits, execute the following command.

[CONTROL 1;3+12+48+192] or [CONTROL 1;255]

Registration number 2

Example :

Right justify LIST output

Execute the following command.

[CONTROL 2;5]

When the LLIST or GLIST command is run, 5 spaces are inserted in front of each line number before output of the list.

```
-----10 PRINT "ADVANTEST"  
-----20 PRINT "   NETWORK"  
-----30 PRINT "       ANALYZER"  
-----40 END
```

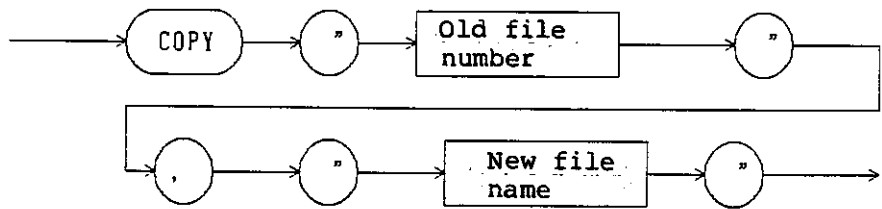
5. COPY

Outline

Copies registered file to floppy disk.

Syntax

(1)-1



(1)-2

COPY "Old file name", "New file name"

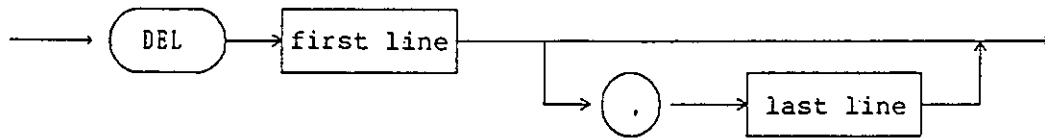
Commentary

- Copy old file name to new file name. No action taken if file name with same name as "new file name" already exists, or if the new file name is the same as the old file name. Both file names can be specified using character string representation.

6. DEL

Outline Deletes line from program.

Syntax (1)-1



(1)-2
DEL First line [, Last line]

Note : Comma (,) may be changed for space.

Commentary

- Delete the program from the First line to the last line.
- Specify any line number from 1 to 65535.
- Deletion is not made if no number is specified.

Example

DEL 10	Deletes only the line number 10.
DEL 10, 100	Deletes the lines from 10 to 100.
DEL , 100	Deletes the program from the first line to the line number 100.
DEL 10,	Deletes the program from the line number 10 to the last line.

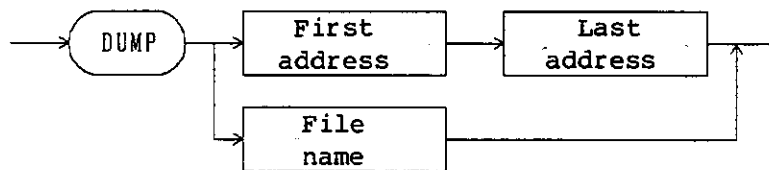
7. DUMP

Outline

Displays memory and files.

Syntax

(1)-1



(1)-2

DUMP <First address Last address>|<File name>

Commentary

- This debugger command displays the entire memory or file as it.
- When two equations are specified, the system assumes them to be the first address and the last address of the memory, and displays the portion between them in hexadecimal and associated ASCII codes.
- When the character string is specified, the system assumes it is the file name and displays the entire file.
- Since display waits for input per page, press the **RETURN** key to display the next page. Press the **RETURN** key after any key to terminate the DUMP command.

Example

DUMP "AFILE"

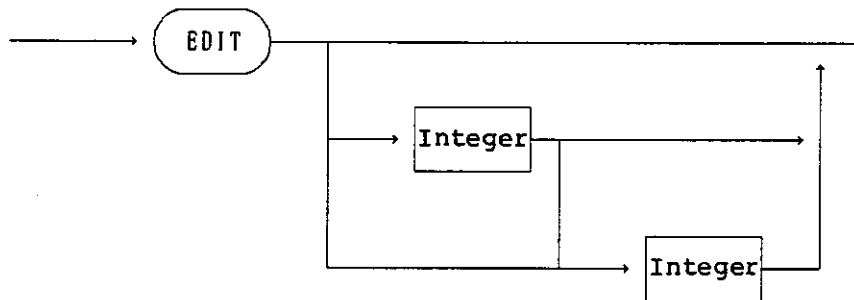
8. EDIT

Outline

Starts program editor mode. During input of program, line numbers appear automatically on the CRT screen.

Syntax

(1)-1



(1)-2

EDIT [[Integer][Integer]]

Note : Specify any integer from 1 to 65535.

Commentary

- Display several lines before and after the current line when program editor mode is started.
- The first integer specifies the start line number, and the second integer specifies the line increment. Both values are valid only when editor mode is started with no program in the BASIC buffer (such as immediately after SCRATCH).

EDIT Start line number Increment

These integer numbers can be omitted, defaults values of 10 being set automatically for each integer.

Example

EDIT
EDIT 100
EDIT 30 5

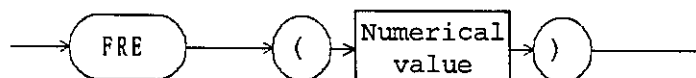
9. FRE

Outline

Indicates the remaining memory capacity for the BASIC program.

Syntax

(1)-1



(1)-2

FRE(Numerical value)

Commentary

This system function indicates in alphanumerics the approximate remaining memory capacity for the BASIC program.

The system only makes a rough judgment without reconstructing the memory, thus once saved, the indicates capacity may be larger than the real capacity.

Example

PRINT FRE (0)

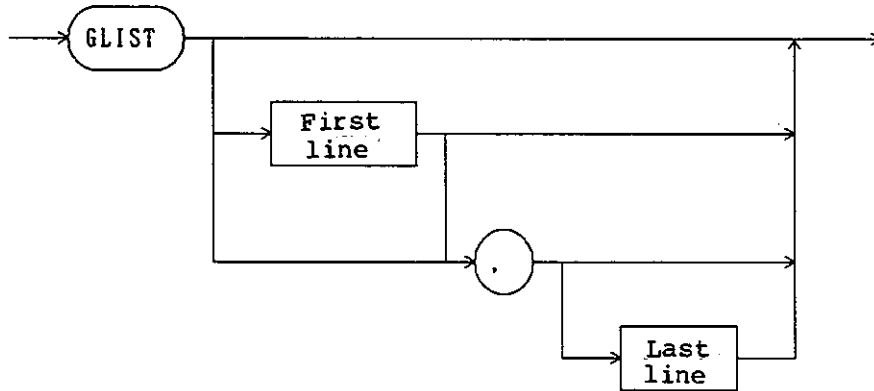
10. GLIST

Outline

Outputs of program list to printer etc. via GPIB.

Syntax

(1)-1



(1)-2

- GLIST First line [,] Last line ①
- or
- GLIST First line, ②
- or
- GLIST First line ③
- or
- GLIST, Last line ④
- or
- GLIST, ⑤
- or
- GLIST ⑥

Note: Specify first line and last line with any integer from 1 to 65535.

Commentary

- Output of BASIC program list to printer etc. connected to GPIB.
- The printer GPIB address is set by PRINTER statement.
- (See the Syntax.)
 - ① Displays the portion specified by the first line and the last line.
 - ② Displays the portion specified by the first line and comma, where the comma represents the last line of the program. Display continues up to the last line, though not specified.
 - ③ Displays only the first line.
 - ④ The first line is omitted, and displays the actual first line of the program to the specified last line. Comma cannot be omitted.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.3 BASIC Command Syntax

- ⑤ ⑥ When both the first line and the last line are omitted, all the lines are displayed.

Example

```
GLIST  
GLIST 100, 200
```

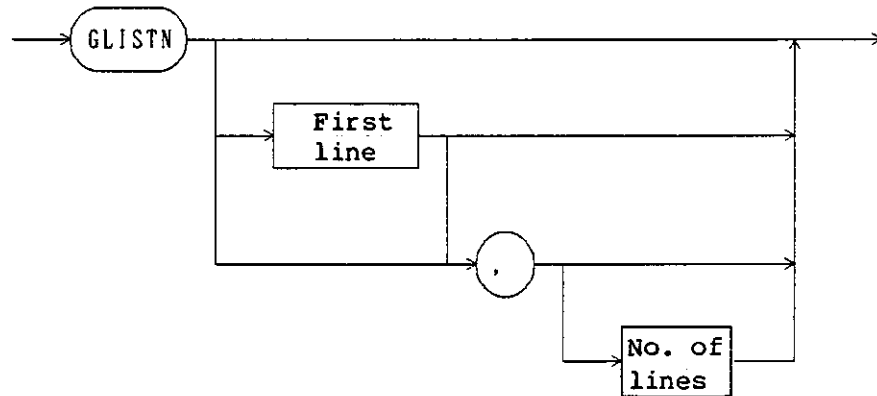
11. GLISTN

Outline

Outputs of program list to printer etc. via GPIB.

Syntax

(1)-1



(1)-2

- GLISTN first line [,] Number of line ①
- or
- GLISTN first line, ②
- or
- GLISTN first line ③
- or
- GLISTN, Number of lines ④
- or
- GLISTN, ⑤
- or
- GLISTN ⑥

Commentary

- Output of BASIC program list to printer etc. connected to GPIB.
- The printer GPIB address is set by PRINTER statement.
- Output program list of the number of lines specified at number of lines starting from the line number specified at first line.
- If the number of lines is a negative value, the number of lines counting in reverse is listed.
- (See the syntax)

- ① Outputs the specified number of lines counting from the first line. When the specified number of lines has a negative value, the count is reversed.
- ② The number of lines is omitted. Outputs the portion specified by the first line and the last line. The system assumes method ③ if the required comma is omitted.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.3 BASIC Command Syntax

- ③ Outputs only the first line.
- ④ The first line is not specified. If the specified number of lines has a positive value, output starts from the first line, and if the specified number of lines has a negative value, the output is reversed from the last line.
- ⑤ ⑥ When the specification is the comma only, without parameters, all the lines are output.

Example

```
GLISTN  
GLISTN 100 20  
GLISTN 200,-10
```

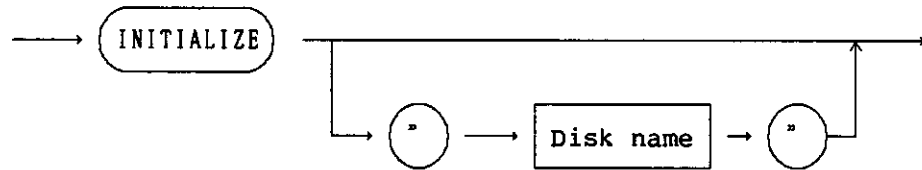
12. INITIALIZE (INIT)

Outline

Initializes a new floppy disk, or a floppy disk whose content is no longer required.

Syntax

(1)-1



(1)-2

INITIALIZE ["Disk name"]

Commentary

o Floppy disks used in the network analyzer must first be initialized by an initialization process specific for the network analyzer. A disk name used to identify the disk is input at this stage. If no disk name is set, the disk name automatically becomes 'ADVANTEST : NA'. This disk name can be specified as a character string expression.

Note

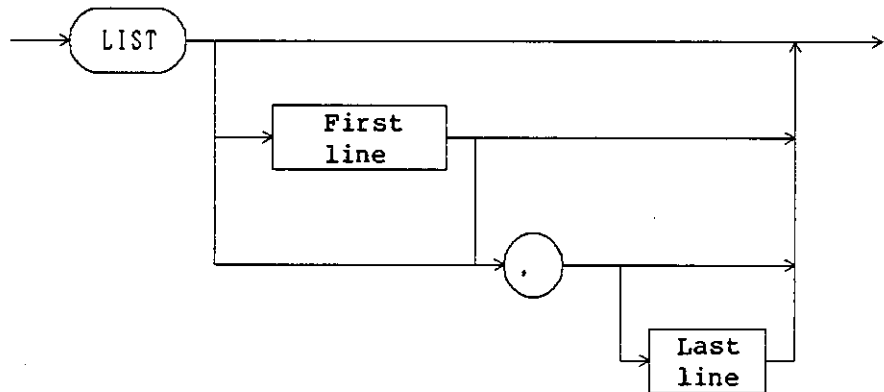
Disk names may contain up to 16 characters, and the character which may be used are the same as those which may be used in file names. (See 24. SAVE Note.)

13. LIST

Outline
Syntax

Displays program list on CRT screen.

(1)-1



(1)-2

- LIST first line [,] last line ①
- or
- LIST first line, ②
- or
- LIST first line ③
- or
- LIST, last line ④
- or
- LIST, ⑤
- or
- LIST ⑥

Note : When the numerical value for the first line or that of the last line is specified, the system assumes the first line.
Specify any integer from 1 to 65535.

Commentary

The portion of BASIC program list specified by the parameter is displayed on the CRT screen. Displaying of list can be interrupted by the stop key. Unlike program execution, resumption of display from the point of interruption is impossible.

Line numbers are specified by equations. Line number zero and number 65536 or higher are given special meanings, the first line and the last line of the program. A line number that is lower or higher than the actual program line number in the buffer is also considered the first line and the last line of the program.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.3 BASIC Command Syntax

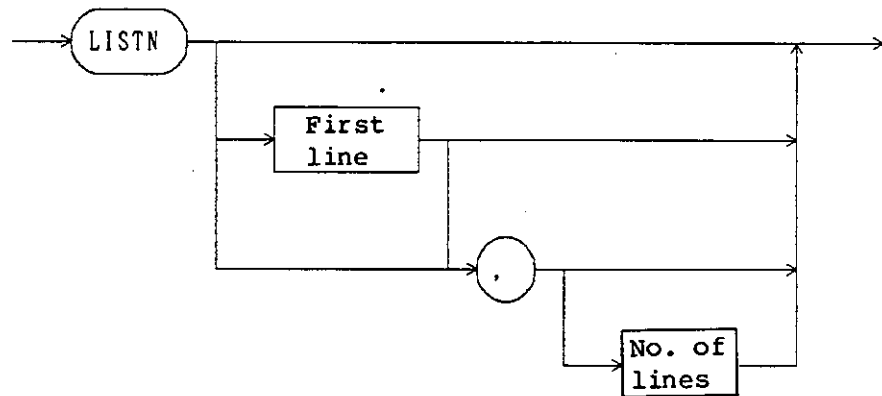
To state the portion to be displayed, use one of the 6 methods listed below. (See the Syntax.)

- ① Displays the portion specified by the first line and the last line.
- ② Displays the portion specified by the first line and comma, where the comma represents the last line of the program. Display continues up to the last line, though not specified.
- ③ Displays only the first line.
- ④ The first line is omitted, and displays the actual first line of the program to the specified last line. Comma cannot be omitted.
- ⑤ ⑥ When both the first line and the last line are omitted, all the lines are displayed.

14. LISTN

Outline Displays program list on CRT screen.

Syntax (1)-1



- (1)-2
- LISTN first line [,] number of lines ①
 - or
 - LISTN first line, ②
 - or
 - LISTN first line ③
 - or
 - LISTN, number of lines ④
 - or
 - LISTN, ⑤
 - or
 - LISTN ⑥

Note: Specify first line and last line with any integer from 1 to 65535.

Commentary The portion of the BASIC program list specified by the parameter is displayed on the CRT screen. In this function, which is basically the same as the LIST command, but the second parameter is the number of lines to be displayed. (See the Syntax.)

- ① Displays the specified number of lines counting from the first line. When the specified number of lines has a negative value, the count is reversed.
- ② The number of lines is omitted. Displays the portion specified by the first line and the last line. The system assumes method ③ if the required comma is omitted.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.3 BASIC Command Syntax

- ③ Displays only the first line.
- ④ The first line is not specified. If the specified number of lines has a positive value, display starts from the first line, and if the specified number of lines has a negative value, the display is reversed from the last line.
- ⑤ ⑥ When the specification is the comma only, without parameters, all the lines are displayed.

Example

```
LISTN  
LISTN 100 20  
LISTN 200,-10
```

Note

In BASIC command patterns apart from EDIT, either character string variables or numerical value representation can be specified. That is, numerical variables used in BASIC can also be used here. For easier reading purposes, however, integer and character string expressions are used in the following pages. The decimal places of real numbers are rounded off to the nearest whole number.

As a rule, commas (,) are not required if the boundary between successive expressions in a BASIC command can be detected in terms of command syntax.

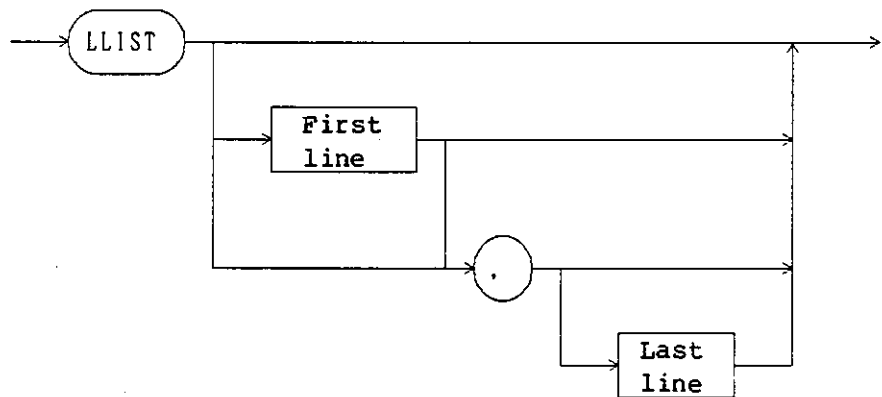
For example, no comma is required in line 2 of the above example since the numeric values 100 and 20 can be read. But in line 3, omission of the comma results in the numeric values being read as 200 minus 10 equals to 190. That is, line 190 would be displayed instead of the ten lines counting back from line 200.

15. LLIST

- Outline
- Syntax

Output of program list to printer etc. via serial port.

(1)-1



(1)-2

- LLIST first line [,] last line ①
- or
- LLIST first line, ②
- or
- LLIST first line ③
- or
- LLIST, last line ④
- or
- LLIST, ⑤
- or
- LLIST ⑥

Note : Specify any integer from 1 to 65535.

Commentary

- Output of BASIC program list to printer etc. connected to the serial port.
- (See the Syntax.)
 - ① Outputs the portion specified by the first line and the last line.
 - ② Outputs the portion specified by the first line and comma, where the comma represents the last line of the program. Output continues up to the last line, though not specified.
 - ③ Outputs only the first line.
 - ④ The first line is omitted, and outputs the first line of the program to the specified last line. Comma cannot be omitted.
 - ⑤ ⑥ When both the first line and the last line are omitted, all the lines are output.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.3 BASIC Command Syntax

Example

```
LLIST  
LLIST 100,200
```

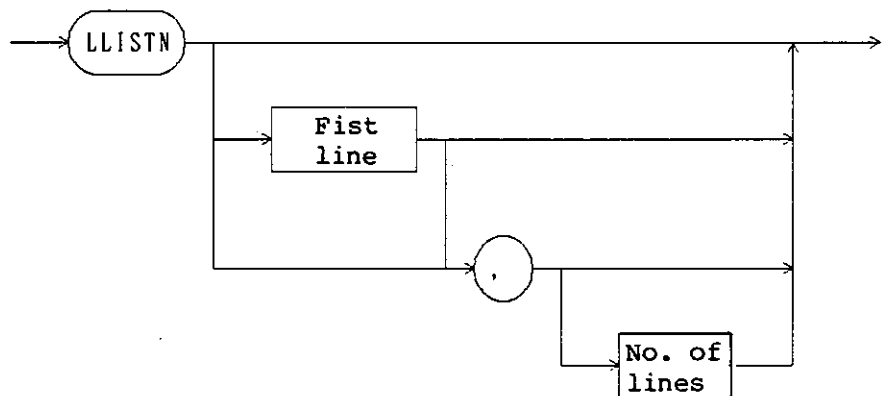
16. LLISTN

Outline

Output of program list to printer etc. via serial port.

Syntax

(1)-1



(1)-2

- LLISTN First line [,] Number of lines ①
- or
- LLISTN First line, ②
- or
- LLISTN First line ③
- or
- LLISTN, Number of lines ④
- or
- LLISTN, ⑤
- or
- LLISTN ⑥

Commentary

- Output of BASIC program list to printer etc. connected to the serial port.
- Output program list of the number of lines specified at number of lines starting from the line number specified at first line.
- If the number of line is a negative value, the number of lines counting in reverse is listed.
- (See the Syntax.)
 - ① Outputs the specified number of lines counting from the first line. When the specified number of lines has a negative value, the count is reversed.
 - ② The number of lines is omitted. Outputs the portion specified by the first line and the last line. The system assumes method ③ if the required comma is omitted.
 - ③ Outputs only the first line.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.3 BASIC Command Syntax

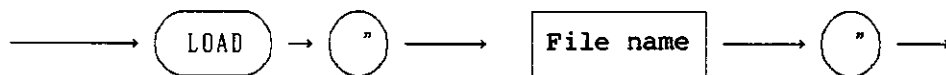
- ④ The first line is not specified. If the specified number of lines has a positive value, output starts from the first line, and if the specified number of lines has a negative value, the output is reversed from the last line.
- ⑤ ⑥ When the specification is the comma only, without parameters, all the lines are output.

Example

```
LLISTN  
LLISTN 100,20  
LLISTN 200,-10
```

17. LOAD

Outline	Calls file from floppy disk.
Syntax	(1)-1



(1)-2
LOAD "File name"

Commentary	Call the file specified by file name to enable editing of that file. Non-BASIC files which cannot be edited (such as system files) cannot be called.
------------	--

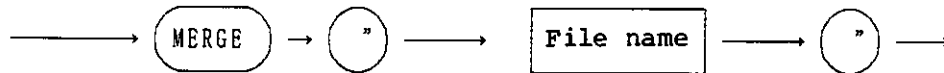
NETWORK ANALYZER
PROGRAMMING MANUAL

5.3 BASIC Command Syntax

18. MERGE

Outline Call file from floppy disk.

Syntax (1)-1



(1)-2
MERGE "File name"

Commentary Unlike LOAD, the BASIC buffer is not initialized prior to loading. The program already present in the BASIC buffer is not cleared unless line numbers coincide. Combination of SCRATCH and MERGE has same function as LOAD.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.3 BASIC Command Syntax

19. PRINTER See [2.7 PRINTER statement] in section 5.4 for details.

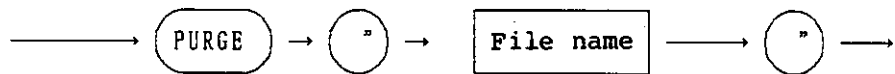
20. PURGE

Outline

Erase file from floppy disk.

Syntax

(1)-1



(1)-2

PURGE "File name"

Commentary

- Erase existing files which are no longer required.
- File names stored by SAVE/RECALL but no longer required can be erased by this command.

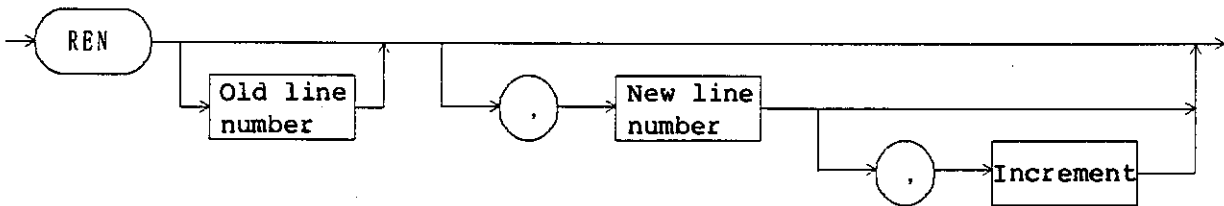
21. REN

Outline

Renumber the line numbers of each program line.

Syntax

(1)-1



(1)-2

REN [[Old line number] [, <New line number> [, <Increment>]]]

Note : Old and new line numbers, and increment, are all integers (1 thru 65535).

The default value for the new line number and increment is 10.

If the old line number is omitted, a comma must be inserted before the new line number to identify that number.

Comma (,) may be changed for space.

Commentary

- The "old line number" is the current program line number where line renumbering is to commence.
- The "new line number" is the new start line number.
- The "increment" is the new line number increment.
- The REN command also changes line numbers used by GOTO, GOSUB etc.
- The REN command cannot generate line numbers greater than 65535. Nor is it possible to change the order of line numbers.

Example

REN Program starts from line 10, and is incremented throughout in steps of 10.
REN 30,50,3 Line number 30 is changed to 50, and subsequent lines are incremented in steps of 3.

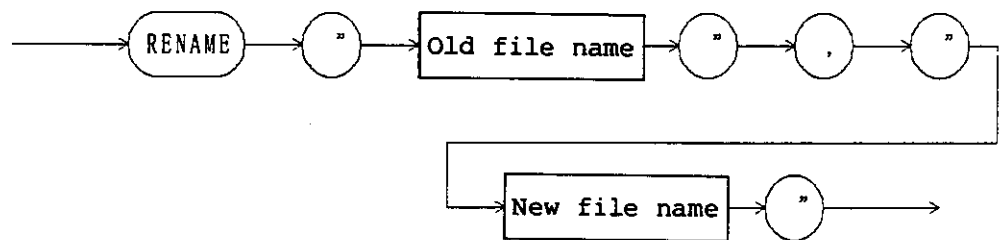
22. RENAME

Outline

Change the name of file stored on floppy disk.

Syntax

(1)-1



(1)-2

RENAME "Old file name", "New file name"

Commentary

- Change old file to new file name. The new file name must not be the same as any existing file name nor the old file name. And since only the name is changed, the contents of the new file are identical to the contents of the old file.

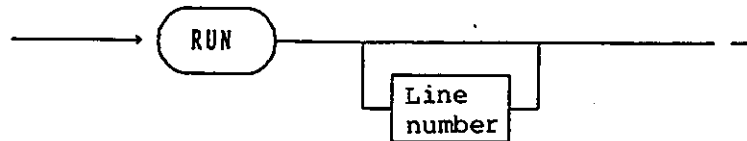
23. RUN

Outline

Run a BASIC program.

Syntax

(1)-1



(1)-2

RUN [Line number]

Commentary

- Run BASIC program from specified line.
- Run program from first line if no line is specified.
- When the RUN command is executed, all variables are cleared prior to commencement, and array declarations etc. are reset.

Example

RUN
RUN 200

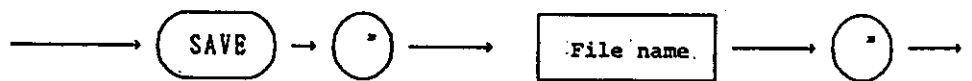
24. SAVE

Outline

Save file to floppy disk.

Syntax

(1)-1



(1)-2

SAVE "File name"

Commentary

- An edited program (from the first statement with a line number up to the last) is registered as a file under the specified file name. If the specified file name already exists, the old file contents are updated by the new file.

Note

File names may consist of up to 16 characters. All characters apart from (double quotation mark) " and space may be used.

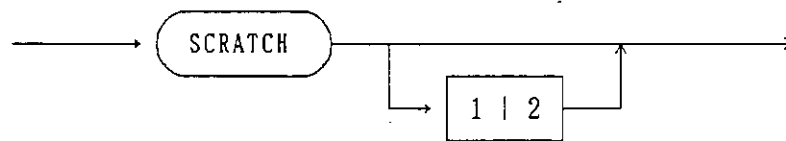
25. SCRATCH

Outline

Erase BASIC program from memory.

Syntax

(1)-1



(1)-2

SCRATCH [1 | 2]

Commentary

- Run this program if the previously loaded BASIC program is no longer required.
- If only the data of the program present in the BASIC buffer is to be initialized, specify 1.
- If only the procedure of the program present in the BASIC buffer is to be initialized, specify 2.

Example

SCRATCH
SCRATCH 1
SCRATCH 2

NETWORK ANALYZER
PROGRAMMING MANUAL

5.3 BASIC Command Syntax

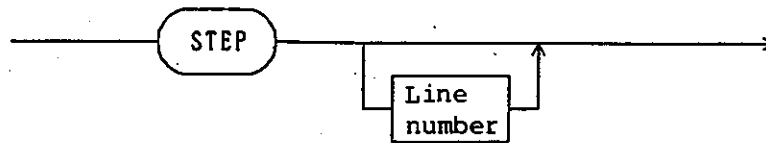
26. STEP

Outline

Run a single line of a BASIC program.

Syntax

(1)-1



(1)-2

STEP [Line number]

Commentary

- Run the single specified line of a BASIC program. Note that STEP cannot be run in a FOR statement.
- Execute the next line after the last executed line if no line is specified.

Example

STEP
STEP 100

5.4 BASIC Statement Syntax

This section describes the following statements in order.

1. BUZZER
2. CLS
3. CURSOR
4. DATA
5. DIM
6. DISABLE INTR
7. ENABLE INTR
8. ERRM\$
9. ERRN
10. FOR-TO-STEP
NEXT
11. GOSUB
RETURN
12. GOTO
13. GPRINT
LPRINT
14. IF THEN
15. INPUT
16. INTEGER
17. LET
18. OFF KEY
19. OFF SRQ
OFF ISRQ
20. ON ERROR
21. ON KEY
22. ON SRQ
ON ISRQ
23. PAUSE
24. PEEK
25. POKE
26. PRINT
27. PRINTER
28. PRINTF
29. READ
30. REM
31. RESTORE
32. SPRINTF
33. SELECT
CASE

NETWORK ANALYZER
PROGRAMMING MANUAL

5.4 BASIC Statement Syntax

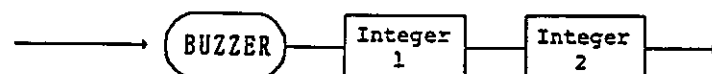
1. BUZZER

Outline

Activate buzzer.

Syntax

(1)-1



(1)-2

BUZZER Integer 1 Integer 2

Commentary

- When BUZZER statement is executed, the network analyzer built-in buzzer is activated in accordance with the designation.
- The buzzer tone is specified by integer 1. Specify any value from 0 (high tone) to 255 (low tone). Integer 2 shows time (unit: ms).

Example

```
10 FOR I = 1 TO 255  
20 BUZZER I, 10  
30 NEXT I  
40 STOP
```

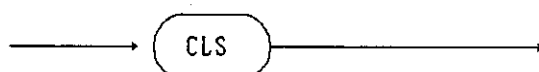
2. CLS

Outline

Clear the CRT screen.

Syntax

(1)-1



(1)-2
CLS

Commentary

- Clear all characters displayed on the CRT screen.
- At the same time that the screen is cleared, the cursor is returned to the home position.

Example

10 CLS

NETWORK ANALYZER
PROGRAMMING MANUAL

5.4 BASIC Statement Syntax

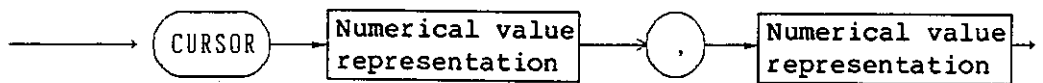
3. CURSOR

Outline

Move cursor to specified coordinate position.

Syntax

(1)-1



(1)-2

CURSOR Numerical value representation, Numerical value representation

Note : Numerical value representation:

X axis designation, column direction

Numerical value representation:

Y axis designation, Row direction

Comma (,) may be changed for space.

Commentary

- Move cursor to specified position on the CRT screen.
- The first value enclosed in parentheses indicates the X axis coordinate, and the second value indicates the Y axis coordinate.

CURSOR X axis coordinate, Y axis coordinate

These two values must lie within the following ranges.

$0 \leq X \text{ axis coordinate} \leq 45$

$0 \leq Y \text{ axis coordinate} \leq 24$

Example

```
10 PRINT CHR$(12)
20 X=0:Y=4:X1=1:Y1=1
30 CURSOR X,Y:PRINT "*"
40 X=X+X1:Y=Y+Y1
50 IF X<=0 OR 46<=X THEN X1 *= -1
60 IF Y<=0 OR 26<=Y THEN Y1 *= -1
70 CURSOR X,Y:PRINT ""
80 GOTO 30
90 STOP
```

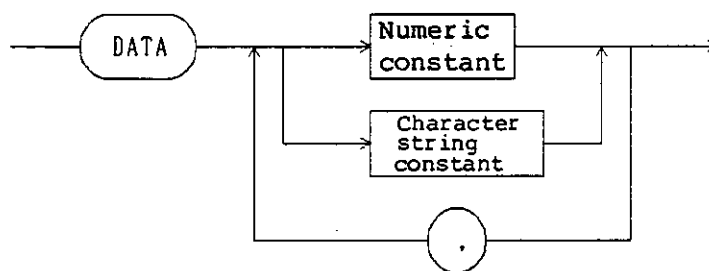
4. DATA

Outline

Defines numeric values and character strings to be read by the READ statement.

Syntax

(1)-1



(1)-2

DATA Numeric constant | character string constant
{,Numeric constant | character string constant}

Commentary

DATA statements are not executed but read by the READ statement.

Therefore though the DATA statements can be at any line number, they must be arranged in the order of reference.

To rearrange them, use the RESTORE statement.

More than one constant, separated by commas (,), can be specified in a single DATA statement. Put character strings in double quotations (") as a character string constant.

Note

Parameters in the DATA statement cannot contain equations with variables.

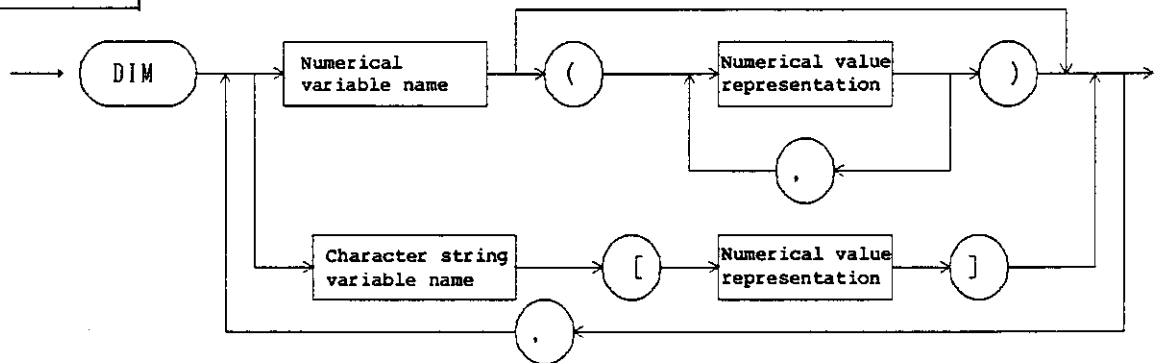
5. DIM

Outline

Array variable or character string variable definition declaration.

Syntax

(1)-1



(1)-2

DIM<A> | {,<A> | }

Note : A: Numerical variable name [(Numerical value representation {,Numerical value representation})]
B: Character string variable name [Numerical value representation]

Commentary

- When an array variable or character string variable is used, the array variable name and array size must be defined by DIM statement. If name and size are not defined, the array becomes 10 elements in 1-dimension, and the character string takes a length of 18 characters.
- When an array is declared using the DIM statement, the array variable of the specified size is stored in memory. Therefore, if the declared variable is too big, there will be insufficient space left for the BASIC program. (An error is generated and program execution is stopped if the array size is greater than the memory space.) (Out of memory)
- If the result of operation on a numerical value representation for array variable size is a real number expression, the decimal places are rounded off to an integer number expression.
- When using a character string variable, the length of the character string is declared by numerical value representation.
- An array variable can't use 0.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.4 BASIC Statement Syntax

Example

10 DIM N(5)	<Execution result>
20 FOR I = 1 TO 5	0.5
30 N(I) = I*I/2	2.0
40 NEXT I	4.5
50 FOR I = 1 TO 5	8.0
60 PRINT N (I)	12.5
70 NEXT I	

6. DISABLE INTR

Outline

Disable acceptance of interrupts.

Syntax

(1)-1



(1)-2

DISABLE INTR

Commentary

- Disable interrupts enabled by ENABLE INTR.
- To enable interrupts again after executing this statement, execute the ENABLE INTR statement. Branch conditions set by ON XXX statement are maintained unchanged in this case. If the interrupt branch conditions are to be changed, use the ON XXX or OFF XXX statement before executing the ENABLE INTR statement.
- Interrupts are disabled from immediately after execution of this program until the ENABLE INTR statement is executed.

Example

```
10 OUTPUT 31; "EDITOFF SRQE"  
20 ON ISRQ GOTO 60  
30 ENABLE INTR  
40 ! LOOP  
50 GOTO 40  
60 DISABLE INTR  
70 PRINT "INTERRUPT"  
80 GOTO
```

7. ENABLE INTR

Outline

Cancel interrupt disable status generated by ON XXX statement or DISABLE INTR.

Syntax

(1)-1



(1)-2

ENABLE INTR

Commentary

- If branching is generated by interrupt enabled by ON XXX statement, all interrupt generated branching is disabled temporarily. This is to prevent nesting of interrupt processing in cases where another interrupt is generated while a previous interrupt is being processed.
- If this statement is executed when interrupts are enabled again after branching generated by an interrupt has been processed, the interrupt disabled status is cancelled to enable branching by interrupt again.
- If interrupt processing is placed in a subroutine, execution of the processing can be made smoother by inserting this statement immediately before the RETURN statement.
- Also execute this statement if interrupts are to be enabled again after the DISABLE INTR statement is executed.
- Interrupts are disabled from immediately after program execution up to execution of this statement.

Example

```
10 OUTPUT 31; "EDITOFF SRQE"  
20 ON ISRQ GOTO 60  
30 ENABLE INTR  
40 ! LOOP  
50 GOTO 40  
60 DISABLE INTR      ! INTERRUPT  
70 PRINT "INTERRUPT" !  
80 END
```

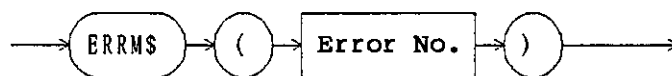

8. ERRM\$

Outline

This statement is a system function for returning an error message of the specified number.

Syntax

(1)-1



(1)-2

ERRM\$(Error No.)

Commentary

The system function returns an error message specified in a parameter.

When specifying 0 as a parameter, it returns the last displayed error message.

The error number structure is as follows:

Error class * 256 + Error message number

Only the error message number is referred to internally although an error number including an error class is specified. Therefore, ERRN can be specified for an error number.

9. ERRN

Outline

This statement is a system variable for retaining an error number.

Syntax

(1)-1



(1)-2
ERRN

Commentary

This is a system variable for retaining an error number generated when the BASIC program is executed.

The system variable is initialized to 0 at the start of the BASIC program and the value is substituted when an error occurs.

The value is initialized to 0 when 0 is substituted explicitly or the BASIC program is reexecuted.

Actual error number structure is as follows:

Error class * 256 + Error message number

Error class

- | | |
|---|--|
| 1 | Associated with the data I/O. |
| 2 | Associated with the data operation. |
| 3 | Associated with the build-in function. |
| 4 | Associated with the BASIC statement. |
| 5 | Others |

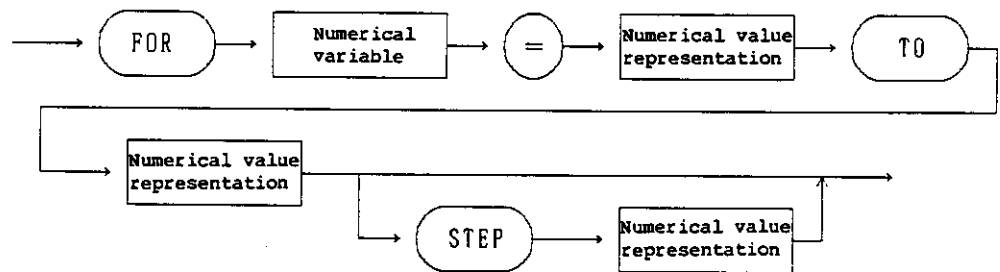
10. FOR-TO-STEP
NEXT

Outline

Program loops are formed by using the FOR and NEXT pair of statements.

Syntax

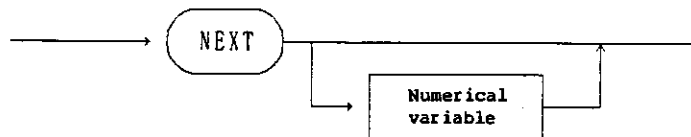
(1)-1



(1)-2

FOR Numerical variable = Numerical value representation
TO Numerical value representation
[STEP Numerical value representation]

(2)-1



(2)-2

NEXT [Numerical variable]

Commentary

- The specified numerical variable is used as a loop counter with changes made one step (increment) at a time from initial to final value. The loop is stopped when the counter value is greater than the final value. Counter increase/decrease is made by the NEXT statement. Therefore, the section of program between the FOR and NEXT statements is processed repeatedly.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.4 BASIC Statement Syntax

- The initial and final values and the increment are specified in the following way.
FOR A = (Initial value) TO (Final value)
STEP (Increment)
- If STEP (increment) is omitted, the increment automatically becomes +1.
- The FOR statement to NEXT statement section can be nested.
- The variable name of the loop counter used with a pair of FOR and NEXT statements must be the same in both statements. An error is generated if the name is different. (NEXT without FOR)
- And if the value of the numerical variable used in the loop counter while processing the program between the FOR and NEXT statements is changed, the repetition processing will not proceed in the normal way.
- If the numerical variable after the NEXT statement is omitted, the value for the previous FOR statement is adopted automatically.
- FOR-NEXT looping can be escaped by BREAK statement.
- The program can be branched by CONTINUE statement from the FOR-NEXT loop to the loop of the next step value.

Example

```
10 FOR R = 11 TO 0 STEP -5
20   FOR I = 0 TO PI STEP PI/180
30     X=SIN(I)*R+23
40     Y=COS(I)*R+15
50     CURSOR X,Y:PRINT "*"
60   NEXT I
70 NEXT R
80 STOP
```

11. GOSUB
RETURN

Outline

Branch to and return from the specified subroutine.

Syntax

(1)-1



(1)-2

GOSUB Numerical value representation | Label expression

(2)-1



(2)-2

RETURN

Commentary

- Transfer process control to subroutine starting from the line number specified by integer or label expression. Return to next statement after the GOSUB statement by using the RETURN statement.
 - Always include the RETURN statement at the end of the subroutine to ensure return to the main program.
 - An error is generated if a RETURN statement is executed without subroutine branching.
 - Since the GOSUB statement to RETURN statement section can be nested, branching to another subroutine from the first subroutine is possible. Too much nesting, however, can use up memory space and result in error.
- If a label expression is used in GOTO or GOSUB, and the corresponding line number does not exist, the
- <<< Undefined line: Enter CORRECT line.>>>
- message appears on that line. No further processing is possible since the branch destination does not exist. Insert the correct line number. If this error message line is deleted accidentally, the value of the GOTO or GOSUB label expression is cleared to 0, and any further attempt to execute the program results in the
- Undefined line
- error message appearing. To enable processing to proceed insert the correct label expression value in the GOTO or GOSUB statement.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.4 BASIC Statement Syntax

Example

```
10 FOR I=1 TO 9
20   GOSUB 60
30   GOSUB *PRT
40 NEXT I
50 STOP
60 ! SUB ROUTINE
70 X = I * I
80 RETURN
90 *PRT ! SUB ROUTINE
100 PRINT I; " * " ;I; " = " ;X
110 RETURN
```

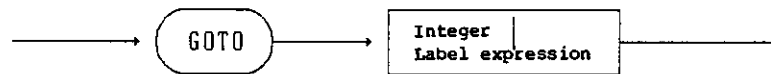
12. GOTO

Outline

Branch to the specified line number.

Syntax

(1)-1



(1)-2

GOTO Integer | Label expression

Commentary

- Branch unconditionally to the specified line number.
- If LIST is executed when the specified line number is found not to exist in the program, a REM statement is automatically inserted in the position corresponding to the missing line number.

Example

```
10 FOR I=1 TO 9
20 GOTO 60
30 GOTO *PRT
40 NEXT I
50 STOP
60 !
70 X = I * I
80 GOTO 30
90 *PRT
100 PRINT I; "*" ;I; "=" ;X
110 GOTO 40
```

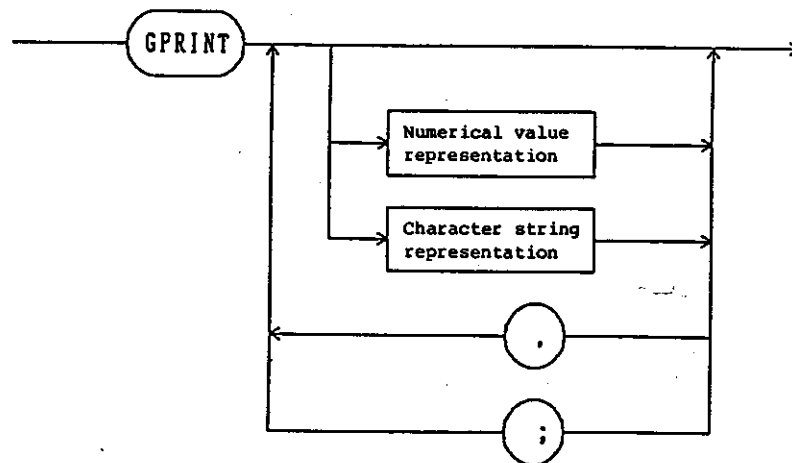
13. GPRINT ; GPIB output
LPRINT ; Serial output

Outline

Output numerical or character string data.

Syntax

(1)-1



(1)-2

```
GPRINT [Numerical value representation  
| Character string representation  
{, ; Numerical value representation |  
Character string representation}]
```

(2)

LPRINT is the same syntax as GPRINT.

Commentary

- Display specified numerical data or character string.
- If numerical values and character strings are partitioned by commas (,) successive values and strings can be output without executing a carriage return.
- And if a comma (,) or semicolon (;) is placed at the end of a GPRINT or LPRINT statement, there is no carriage return at the end of the printer output. Therefore, printing is continued on the same line as the last printing when the next GPRINT or LPRINT statement is executed.

Example

```
100 PRINTER 1  
110 FOR I=0 TO 20  
120 GPRINT I  
130 LPRINT I  
140 NEXT I  
150 STOP
```

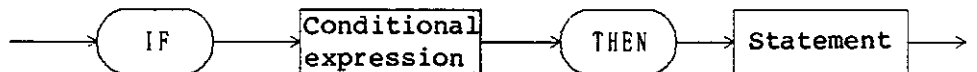

14. IF THEN

Outline

Branch to and execute the specified statement depending on conditions.

Syntax

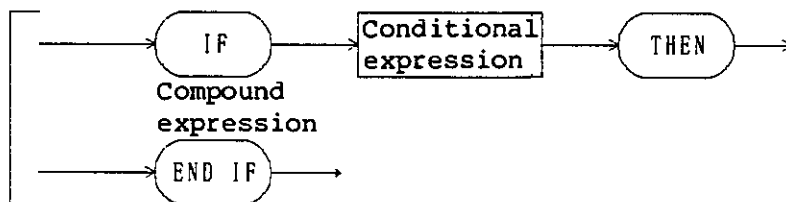
(1)-1



(1)-2

IF <Conditional expression> THEN <Statement>

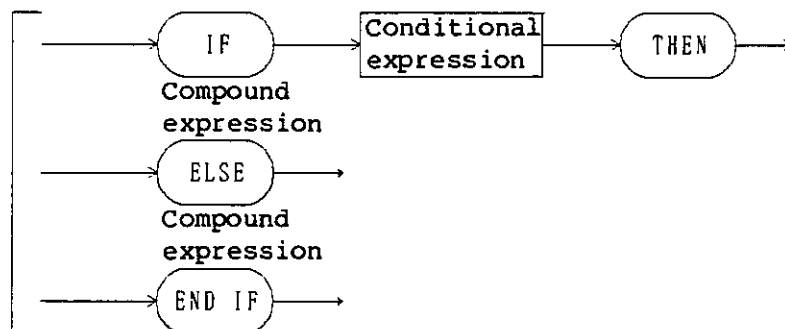
(2)-1



(2)-2

IF <Conditional expression> THEN
Compound expression
END IF

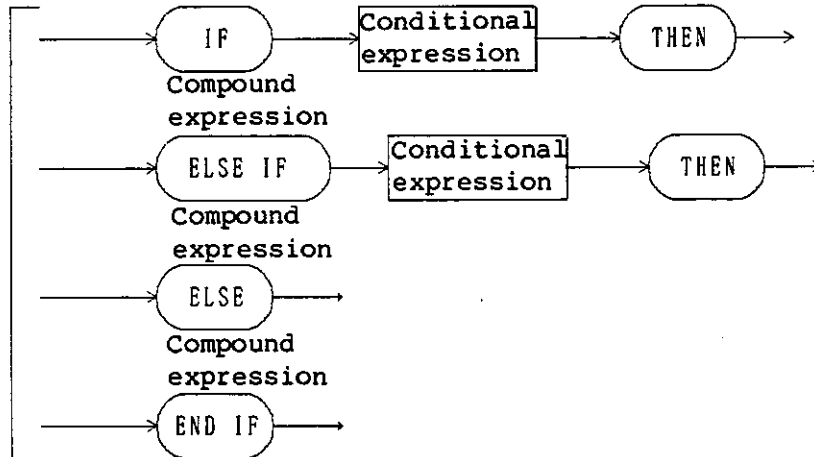
(3)-1



(3)-2

IF <Conditional expression> THEN
Compound expression
ELSE
Compound expression
END IF

(4)-1



(4)-2

```
IF <Conditional expression > THEN
  Compound expression
ELSE IF
  Compound expression
ELSE
  Compound expression
END IF
```

Commentary

- Although the conditional expression is a logical expression, a numerical value representation can also be written here apart from logical expressions using comparison operators. In this case, the operation result is false only if the value is 0, but true if any other value.
- The program is branched and processed according to the logical expression conditions.
- The THEN statement is executed once the logical expression relationship is established. The THEN statement can include successive statements, followed by execution of the next statement.
- If the logical expression relationship is not established, the next line is processed.
- The following six types of logical operators can be used.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.4 BASIC Statement Syntax

A=B (A==B)	Established if A and B are equal
A>B	Established if A is greater than B
A<B	Established if A is smaller than B
A>=B	Established if A is equal to or greater than B
A<=B	Established if A is equal to or smaller than B
A<>B (A!=B)	Established if A and B are not equal

Expressions in parentheses can also be used.

In the above logical expressions, both A and B may be numerical value representations. And numerical value representations can be compared with character string expressions.

Example

```
10 FLG = 0
20 FOR I =0 TO
30 PRINT I;
40 IF (I % 2) =0 THEN FLG = 1
50 IF FLG = 1 THEN
60 PRINT " EVEN" ;
70 FLG = 0
80 END IF
90 PRINT
100 NEXT I
110 STOP
```

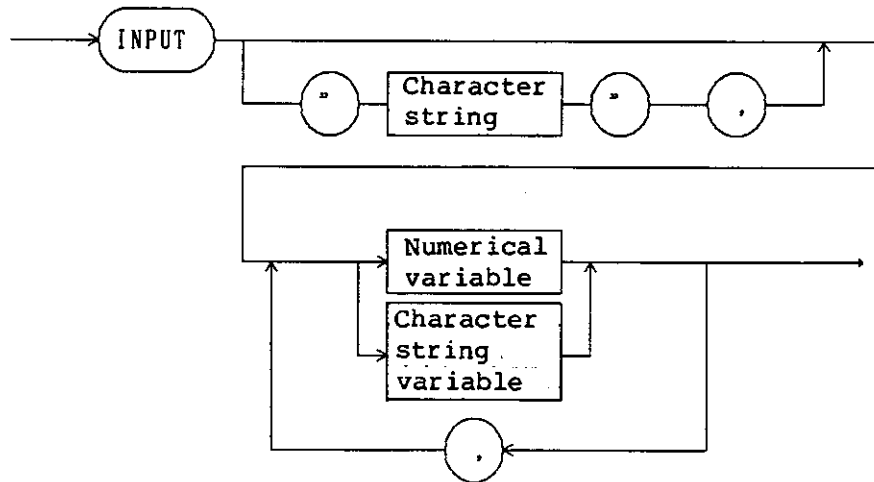
15. INPUT

Outline

Substitute keyboard input data in numerical variable.

Syntax

(1)-1



(1)-2

```
INPUT ["<Character string>",<br>
      Numerical variable | Character string<br>
      {Numerical variable | Character string}]
```

Commentary

- When the INPUT statement is executed, the program is stopped temporarily to wait for input of data from the keyboard. This input wait status is maintained until the ENTER key is pressed, resulting in the key input data being substituted in a variable.
- The INPUT statement can handle both numerical and character string variables. However, if the input contains non-numerical characters (such as alphabetic characters and symbols), all non-numerical characters are disregarded. And if there are no numerical characters at all, a value of 0 is substituted in the variable. No substitution takes place if only the ENTER key is pressed. That is, the value prior to input remains unchanged.
- Character constant inputs do not have to be enclosed between quotation marks.
If register 6 of CONTROL command is set to 1, a function key can be received regardless of input wait.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.4 BASIC Statement Syntax

Example

```
10 OUTPUT 31; "SINGLE EDITON"  
20 INPUT "CENTER FREQUENCY(MHz) ?" ,CF  
30 INPUT "SPAN FREQUENCY(KHz) ?" ,SF  
40 OUTPUT 31; "EDITOFF"  
50 OUTPUT 31; "CENTERF" ,CF, "MHZ"  
60 OUTPUT 31; "SPANF" ,SF, "KHZ"  
70 OUTPUT 31; "SINGLE"  
80 OUTPUT 31; "MAXSRCH"  
90 OUTPUT 31; "MAXSRCH ?"  
100 ENTER 31; F,L,D1,D2  
110 OUTPUT 31; "EDITON"  
120 PRINT "MAX = " ,L  
130 STOP
```

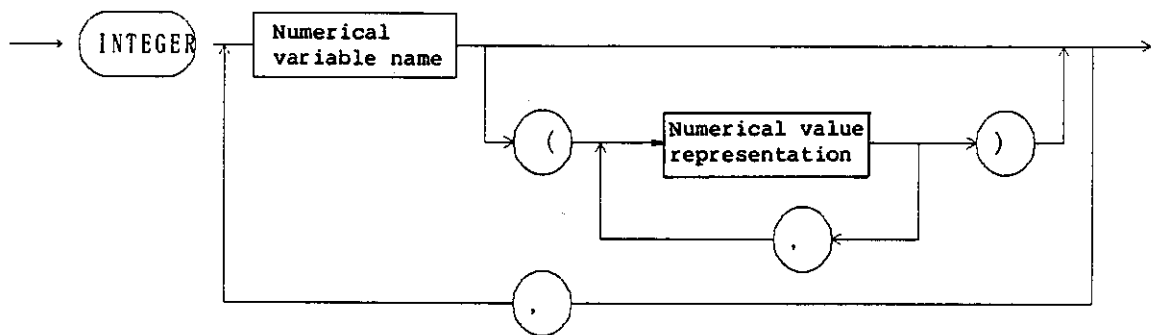
16. INTEGER

Outline

Declaration that the variable or array variable is an integer.

Syntax

(1)-1



(1)-2

INTEGER <A> [B] { ,<A> [B] }

Note : A : Numerical variable

B : (Numerical variable { ,Numerical variable })

Commentary

- If the numeric variable or array variable is specified in the INTEGER statement, the variable is to be an integer type.
- The value for integral variable is the same as that for integral constant.
-2, 147, 483, 648 to +2, 147, 483, 647
- If the variable for integer only is declared with the INTEGER statement, processing time can be reduced.
- If the array is declared with the INTEGER instruction, specified array variable is stored on memory. If too large array is declared, short memory area causes an error. If an error occurs, the program interrupts. (memory space full)
- If several scripts are specified, the array variable with dimensions for the number is specified. (The number of dimension depends on memory capacity.)

Example

```
10 INTEGER ARRAY(2,3)
20 PRINT "J\I" ;
30 PRINT USING "X,3D,3D,3D" ;1,2,3
40 PRINT " " ;
50 FOR I = 1 TO 2
60   FOR J = 1 TO 3
70     ARRAY(I,J) = I*10 + J
80   NEXT J
90 NEXT I
100 FOR I = 1 TO 2
110 PRINT USING "- ,2D,2X,#" ;I
120   FOR J = 1 TO 3
130     PRINT USING "3D,#" ,ARRAY(I,J)
140   NEXT J
150 NEXT I
```

<Execution result>

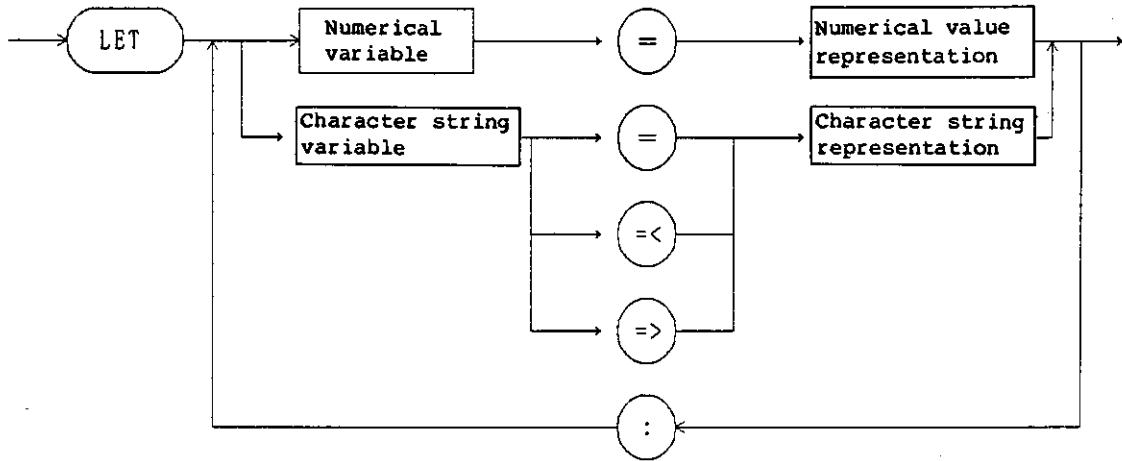
```
J\I 1  2  3
    1  11 12 13
    2  21 22 23
```

17. LET

LET is not used in programs. Direct substitution statements are written.

Outline Substitute in variables.

Syntax (1)-1



(1)-2

LET <A> | { ; <A> | }

Note : A : Numerical variable = Numerical value representation

B : Character string variable = | = < | = > Character string representation

Commentary

- The "=" sign used here denotes substitution. It is not the mathematical equal sign. If the left hand side of this sign is a numerical value, character strings too can convert and substitute the numerical value section. Especially when substituting a character string, the most that can be substituted is the length of the right hand side when "=" is used. With "=>", however, where the character string on the right hand side may be shorter than the character string on the left hand side, the length is substituted in the left hand side with spaces filling the lead. With "=<", on the other hand, spaces are filled in behind. That is, "=>" and "=<" are valid substitution operators only for character strings.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.4 BASIC Statement Syntax

Example

```
10 DIM STR$
20 PRINT "123456789012345678"
30 STR$ = "ABC" :PRINT STR$
40 STR$ = <"OPQ" :PRINT STR$
50 STR$ = >"XYZ" :PRINT STR$
```

<Execution result>

123456789012345678

ABC

OPQ

XYZ

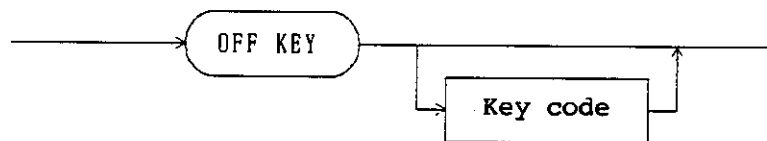
18. OFF KEY

Outline

Cancel branch function and definition by KEY input

Syntax

(1)-1



(1)-2

OFF KEY [Key code]

Commentary

- Cancellation of branching generated by key input interrupt enabled by ON KEY statement.

Example

```
10 ON KEY 2 GOTO 100
20 ENABLE INTR
30 ! LOOP
40 GOTO 30
100 OFF KEY
110 PRINT "OFF KEY"
120 STOP
```

NETWORK ANALYZER
PROGRAMMING MANUAL

5.4 BASIC Statement Syntax

19. OFF SRQ [Enable only in controller mode]
OFF ISRQ

Outline

Cancel branch function and definition by SRQ or ISRQ interrupt.

Syntax

(1)-1

OFF SRQ

(1)-2
OFF SRQ

(2)
OFF ISRQ is the same syntax as OFF SRQ.

Commentary

- Cancellation of branching generated by an interrupt enabled by ON SRQ statement.

Example

```
100 OUTPUT 31; "EDITOFF SRQE"
110 ON ISRQ GOTO *MAX
120 OUTPUT 31; "SINGLE"
130 ENABLE INTR
140 ! LOOP
150 GOTO 140
160 *MAX
170 DISABLE INTR
180 OUTPUT 31; "MAXSRCH"
190 OUTPUT 31; "MAXSRCH?"
200 ENTER 31;F,L,D1,DL2
210 PRINT L
220 GOTO 130
```

Commentary

Address	Details
100	Set measuring screen, and enable SRQ
110	Set internal SRQ interrupt branching
120	Single sweep
130	Accept interrupt
170	Disable interrupt enable SRQ
180	Search for maximum level
190	Request return of maximum level
200	Substitute returned data in respective variables
210	Display level

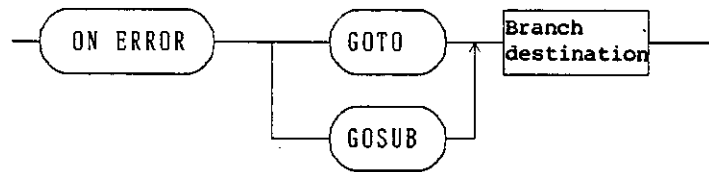
20. ON ERROR

Outline

Specifies branch destination on error.

Syntax

(1)-1



(1)-2

ON ERROR GOTO | GOSUB Branch destination

Commentary

When an error is generated during BASIC program execution, the system displays the line number and error message and stops execution.

In case of built-in function (requests for measuring equipment services) errors, program execution resumes immediately after display of the error message.

To troubleshoot such errors, use the ON ERROR statement. Specify branch destination with a numeric constant, numeric variable, or label. ERRN system variables that keep record of error codes classifies the errors. If unable to recover from the error immediately after generation, use the OFF ERROR statement to avoid resulting in an endless loop.

Example

ON ERROR GOTO 1000

NETWORK ANALYZER
PROGRAMMING MANUAL

5.4 BASIC Statement Syntax

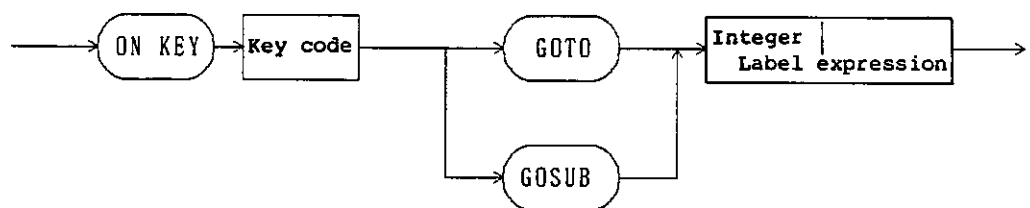
21. ON KEY

Outline

Enable branching by KEY input interrupt.

Syntax

(1)-1



(1)-2

ON KEY Key code GOTO | GOSUB Integer | Label expression

Commentary

- Branch by KEY input interrupt during program execution.
- Branching is executed after the statement being executed when the interrupt was generated has been completed.
- And the return destination required after branching to a subroutine becomes the next statement to be executed after the statement being executed at the time the interrupt was generated.
- Key codes are numerical values from 1 to 6, and correspond to the soft key and function keys F1 thru F6 on the left hand side of the CRT screen.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.4 BASIC Statement Syntax

Example

```
1      CLS
10     ENABLE INTR
20     ON KEY 1 GOTO 1000
30     ON KEY 2 GOTO 1100
40     ON KEY 3 GOTO 1200
50     ON KEY 4 GOTO 1300
60     ON KEY 5 GOTO 1400
70     ON KEY 6 GOTO 1500
75     COUNT = 10
80     *HERE:
85     I = 0: PRINT ""
90     IF I = COUNT THEN GOTO *HERE
100    ++I: PRINT ">" ;
101    GOTO 90
1000   PRINT "FIRST KEY"
1001   COUNT = 1
1010   GOTO *HERE
1100   PRINT "SECOND KEY"
1101   COUNT = 10
1110   GOTO *HERE
1200   PRINT "THIRD KEY"
1201   COUNT = 20
1210   GOTO *HERE
1300   PRINT "FOURTH KEY"
1301   COUNT = 30
1310   GOTO *HERE
1400   PRINT "FIFTH KEY"
1401   COUNT = 40
1410   GOTO *HERE
1500   PRINT "SIXTH KEY"
1501   COUNT = 50
1510   GOTO *HERE
```

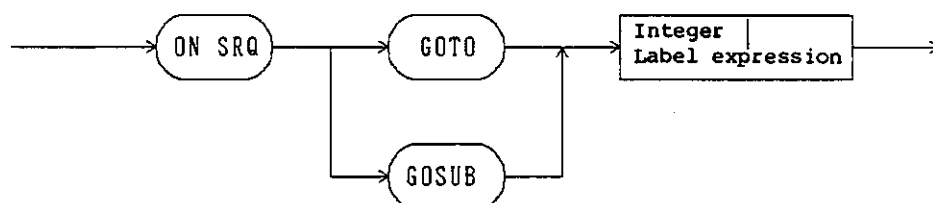
22. ON SRQ [ON SRQ is enable only in controller mode]
ON ISRQ

Outline

Enable interrupt branching by external SRQ signal via GPIB.
(ON SRQ)
Or, enable interrupt branching by when an internal
interrupt source is generated. (ON ISRQ)

Syntax

(1)-1



(1)-2

ON SRQ GOTO | GOSUB Integer | Label expression

Commentary

- Execute branching by interrupt during execution of program.
- Branching is executed after the statement being executed at the time of the interrupt has been completed.
- And the return destination required after branching to a subroutine becomes the next statement to be executed after the statement being executed at the time the interrupt was generated.
- ON SRQ is capable of interrupt branching by SRQ signal from external GPIB only when executing in controller mode.

Example

Search for MAX during each single sweep.

```

100 OUTPUT 31; "EDITOFF"
110 ON ISRQ GOTO *MAX
120 OUTPUT 31; "SRQE"
130 ENABLE INTR
135 OUTPUT 31; "SINGLE"
140 ! LOOP
150 GOTO 140
160 *MAX
170 DISABLE INTR
180 OUTPUT 31; "MAXSRCH"
190 OUTPUT 31; "MAXSRCH?"
200 ENTER 31;F,L,D1,D2
210 PRINT L
220 GOTO 130
  
```

NETWORK ANALYZER
PROGRAMMING MANUAL

5.4 BASIC Statement Syntax

<Commentary>

Address	Details
100	Set measuring screen
110	Set internal SRQ interrupt branching
120	Outputs SRQ when sweep is terminated.
130	Accept interrupt
135	Single sweep
170	Disable interrupt
180	Search for maximum level
190	Request return of maximum level
200	Substitute returned data in respective variables
210	Display level

23. PAUSE

Outline

Temporarily halts program execution.

Syntax

(1)-1



(1)-2

PAUSE

Commentary

This BASIC command temporarily halts the BASIC program execution, thus resumption from the interrupted line by the CONT command is possible.

To halt execution from outside the program, press the **STOP** key.

Example

```
10 FOR I=1 TO 9
20 GOTO 69
30 GOTO *PRT
40 NEXT I
50 PAUSE
60 !
70 X = I * I
80 GOTO 30
90 *PRT
100 PRINT I; "*" ;I; "=" ;X
110 GOTO 40
```

24. PEEK

Outline

The PEEK system function is used for maintenance of this instrument. This function reads the contents of built-in instrument memory.

Syntax

(1)
PEEK (side, address, type)

Note : side : 0 I/O CPU board
 1 Main CPU board
 address : The address from which data is read
 type : 0 Single-byte unit (char)
 1 Two-byte unit (short)
 Others ... Four-byte unit (long)

Commentary

This function is used for maintenance only.
It is not used for ordinary measurement.
The PEEK function reads data from the specified address of the specified board and returns it as the return value.

Example

```
10 side = 0 : I/O CPU board
20 address = 0x5ff80
30 type = 0
40 FOR i = address TO 0x5ffff
50 PRINTF "%c", PEEK(side, i, type)
60 NEXT i
```

25. POKE

Outline

The POKE command is used for maintenance of the equipment. This command writes data in the equipment built-in memory.

Syntax

(1)
POKE side, address, data, type

Note : side : 0 I/O CPU board
 1 main CPU board
 address : The address where the data is written
 data : The data to be written in the specified address
 type : 0 Single-byte unit (char)
 1 Two-byte unit (short)
 Others ... Four-byte unit (long)

Commentary

This function is enabled for maintenance only. It writes data in specified memory address on the specific board in unit specified by 'type'. Thus, the user must have a special knowledge of memory. If the user rewrites important system data, he cannot see what happens. So, this function is not used for normal measurement. To use the function, set control register 5 to 1 (see Item 5.3 [4. CONTROL]).

Example

POKE 0 0x100000 0xFF 0
A single byte of x'FF' is written in address x'100000'.

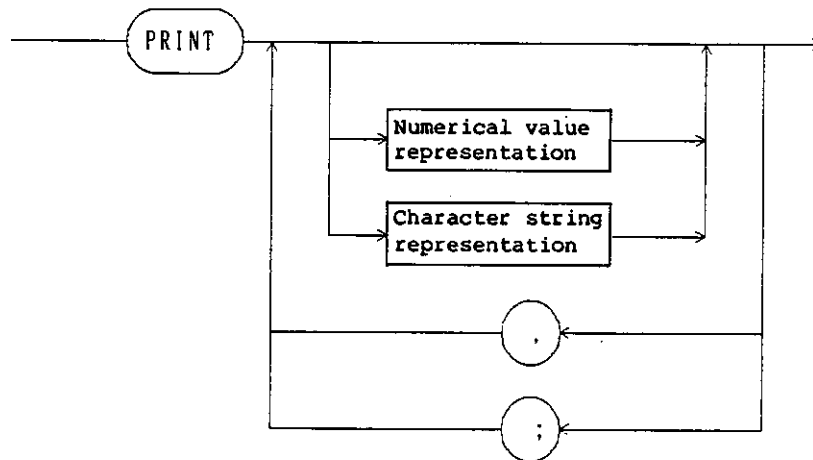
26. PRINT [USING]

Outline

Display numerical or character string data.

Syntax

(1)-1



(1)-2

```
PRINT [Numerical value representation |  
Character string representation  
{,|; Numerical value representation |  
Character string representation}]
```

Commentary

- Display specified numerical data or character string.
- If numerical values and character strings are partitioned by commas (,) successive values and strings can be output without executing a carriage return.
- And if a semicolon (;) is placed at the end of a PRINT statement, output. Therefore, printing is continued on the same line as the last printing when the next PRINT statement is executed.

Example

```
10 PRINT 123*456  
20 PRINT "ABC"  
30 PRINT "Freq.=" , A , "Hz"  
40 PRINT I ,
```

- PRINT USING Format designation
expression; [Expression[...]]

The format designation expression is a character string representation where the format is specified with image specifications partitioned by commas.

Image specifications

- D : Display space in the remaining part of specified field.
- Z : Specify 0 in the remaining part of specified field.
- K : Display expression value without change.
- S : Always append + or - sign flag.
- M : Append - sign flag, or take a space when positive.
- . : Display decimal point.
- E : Display e, sign, and exponential part.
- H : Display expression value without change, but with decimal point displayed in European format.
- R : Display European format decimal point.
- * : Specify * in the remaining part of specified field.
- A : Display single character.
- k : Display character string without change.
- X : Display space.

Literal :

Use \" to enclose sections to be written literally in literal format designation expressions.

- B : Display expression value as ASCII code.
- @ : New page
- + : Shift display position to start of same line.
- : Shift display position to start of next line.
- # : No final carriage return
- n : Repeat of image specifications can be specified with the number.
3D and 2D are the same as DDD.DD. 4A is the same as AAAA.

Example 1

```
10 PRINT USING "4Z,2X,5D,2X,5*" ;123,-444,567
```

Results of execution

```
0123 -444 **567
```

Example 2

```
10 PRINT USING "S3D,X,S3D" ;-4.5,465  
20 PRINT USING "M3Z.Z,X,M3ZR3Z" ;1.26,-5.452
```

Results of execution

```
-4 +465  
001.3 -005,452
```

NETWORK ANALYZER
PROGRAMMING MANUAL

5.4 BASIC Statement Syntax

Example 3

```
10 PRINT USING "K,X,H" ; 5.03884e+22,4.5563
```

```
<Results of execution>  
5.03884e+22 4,5563
```

Example 4

```
10 PRINT USING "k,#" ; "character:"  
20 PRINT USING "B" ;69
```

```
<Results of execution>  
character:E
```

Example 5

```
10 PRINT USING "\ " ..... "\",+,A" ; "*"   
20 PRINT USING "K,-,\ " .END. "\ " ;"string"
```

```
<Results of execution>  
*.....  
string  
.END.
```

Example 6

```
100 PRINT USING "DDD.DD" ;1.2  
110 PRINT USING "ZZZ.ZZ" ;1.2  
120 PRINT USING "K" ;1.2  
130 PRINT USING "SDDD.DD" ;1.2  
140 PRINT USING "MDDD.DD" ;1.2  
150 PRINT USING "MDDD.DD" ;-1.2  
160 PRINT USING "H" ; 1.2  
170 PRINT USING "DDDRDD" ; 1.2  
180 PRINT USING "****.**" ; 1.2  
190 PRINT USING "A" ; "A" ; "a"  
200 PRINT USING "k" ; "string"  
210 PRINT USING "B" , 42  
220 PRINT USING "3D.2D" ;1.2
```

```
<Results of execution>  
1.20  
001.20  
1.2  
+1.20  
1.20  
-1.20  
1,2  
1,20  
**1.20  
a  
string  
*  
1.20
```

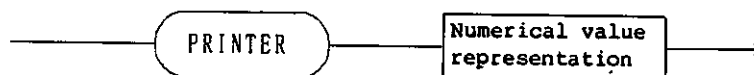
27. PRINTER

Outline

Specify device address to be sent to the printer.

Syntax

(1)-1



(1)-2

PRINTER Numerical value representation

Commentary

- The device address of the printer connected to the GPIB is passed to the network analyzer by this PRINTER command. Before executing a PRINT statement, always specify (in the network analyzer) the printer address by this PRINTER statement.
- The device address is an integer from 0 to 30.

Example

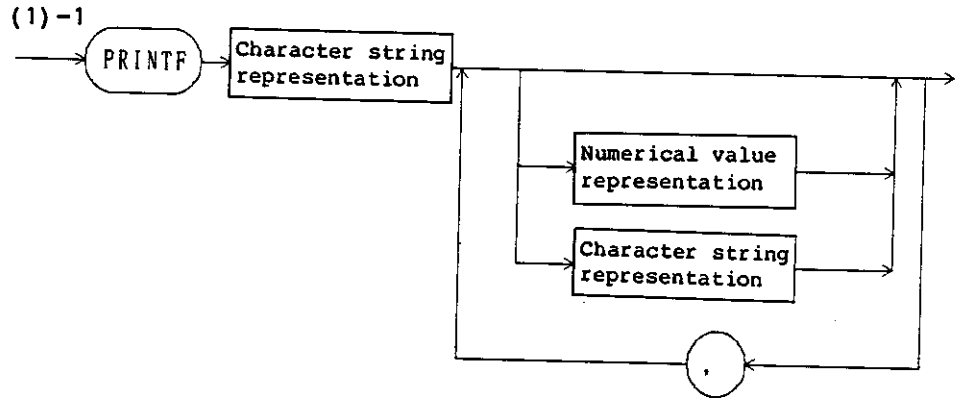
10 PRINTER_1

28. PRINTF

Outline

Display numerical or character string data.

Syntax



(1)-2

PRINTF Character string representation
[Numerical value representation |
Character string representation
{, Numerical value representation |
Character string representation}]

Commentary

- Display specified numerical data or character string.
- If numerical values and character strings are partitioned by commas (,) successive values and strings can be output without executing a carriage return. When a line is fed, specify "\n" in the format.
- The character string representation in the first parameter is used to specify the format of subsequent parameters.

The format designation method is outlined below.

PRINTF Format designation expression; [[Expression[Expression[...]]]]

The format designation method resembles the Printf function in C language.

The format designation expression is a character string type expression, and the output format is specified by the following parameters following %. Other character strings apart from this format are simply straight forward outputs.

If output of % is required, use %%.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.4 BASIC Statement Syntax

% [-], [0], and [.n] characters
- : Left justify within specified field, but right justify if no designation.
0 : Select 0 instead of spaces as the character used to fill up the remainder of the specified field.
m : Take m characters of field.
.n : Output in n-digit precision. When specified for character string, this value becomes the length of the actual character string.
Characters : d : Decimal number with sign
o : Octal number
x : Hexadecimal number
s : Character string
e : Floating decimal point display with sign
f : Floating decimal point display with sign

Example

```
10 N = 500000
20 U = LOG(1+1/N)
30 V = U - 1 / N)
40 PRINTF "%7d %16.5e %16.5e n" ,N,U,V
50 PRINTF "% n" , "end "
```

<Results of execution>

```
50000 2.00000e-06 -1.99994e-12
end
```

NETWORK ANALYZER
PROGRAMMING MANUAL

5.4 BASIC Statement Syntax

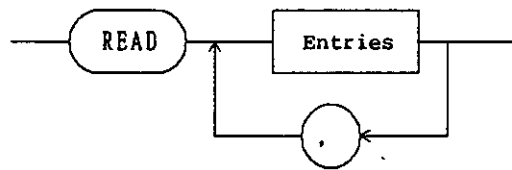
29. READ

Outline

Replaces constants in the DATA statement with variables.

Syntax

(1)-1



(1)-2

READ Entries {,Entries }

Commentary

This statement replaces numerics and character strings defined by the DATA statement with variables specified by the argument.

When the system encounters a READ statement, it searches for the DATA statements.

With the first READ statement, the system starts searching each line number from the head of the program in descending order (if not rearranged by the RESTORE statement) and replaces the first argument found with the variable.

Then the system keeps on searching for DATA statement constants and replaces them one by one.

If the number of constants specified by the DATA statements is less than the number of variables in the READ statements, it results in an error.

In this case, the line numbers of the READ and DATA statements are ignored.

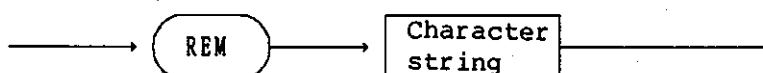
30. REM

Outline

Program remarks.

Syntax

(1)-1



(1)-2

REM <Character string>

Commentary

- Use the REM statement to insert remarks in the program.
- Since the REM statement is not executed, any character string may be inserted after REM. Any alphanumeric character or symbol may be included.
- The REM statement can also be represented by an exclamation mark (!).
- Colons (:) cannot be used for multiple statement purposes after a REM statement. Everything including the colon is regarded as remarks.

Example

```
10 REM "PROGRAM 1"  
20 ! 1983-JUN-02  
30 A=A+1: ! INCREMENT A
```

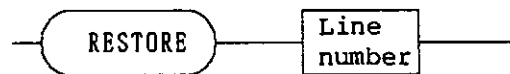
31. RESTORE

Outline

Specifies DATA lines to be read by the next READ statement.

Syntax

(1)-1



(1)-2

RESTORE Line number

Commentary

Specify the line number with an expression or label. If not specified, the DATA statement constants are read from the head of the program to be specified by the next READ statement.

Any line number after the argument line number that is considered the starting position of the search can be specified.

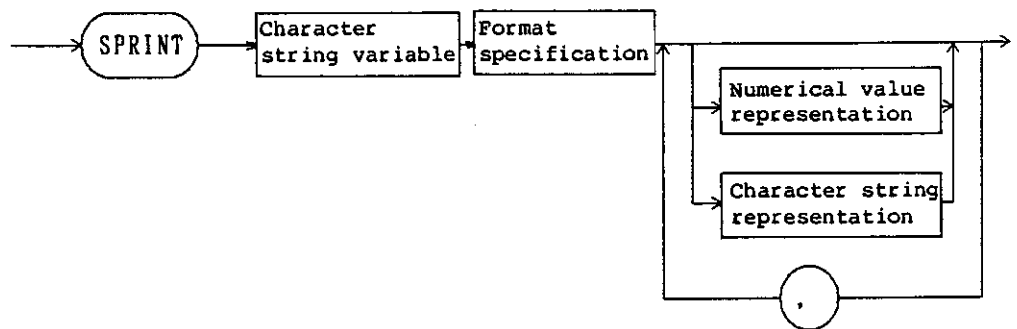
32. SPRINTF

Outline

Convert the format in accordance with the format conversion specification for PRINTF command, and assign the result to the character string variable.

Syntax

(1)-1



(1)-2

SPRINT character string variable format specification
[Numerical value representation |
character string representation]
{, [Numerical value representation |
character string representation]}

Commentary

Covert the value of expression using a method of PRINTF format specification, and assign the result to the character string variable of the first parameter.

Refer to the 'PRINTF' for a method of the format specification. Special attention should be taken to the method of format specification, the number of expression, and character string variable field.

If the character string variable field is not enough to assign the result, a basic buffer will be broken.

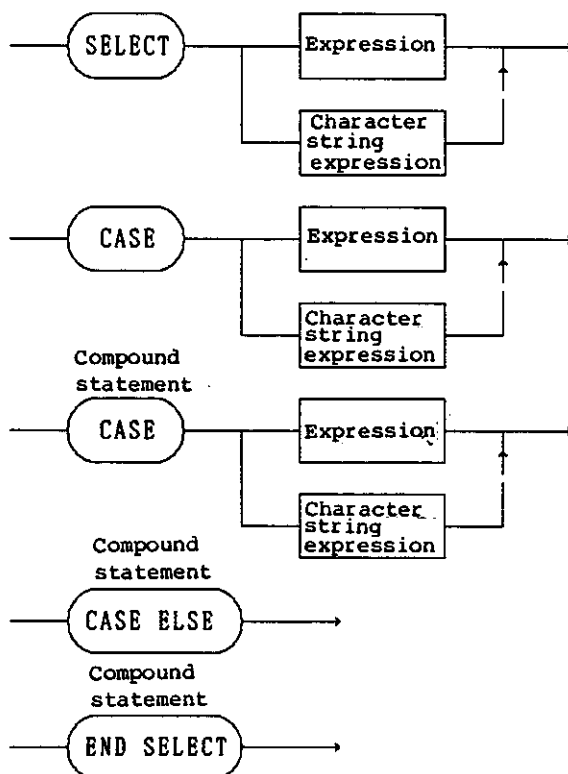
33. SELECT, CASE

Outline

Branch several times using a value in the expression.

Syntax

(1)



Commentary

This statement executes all the compound statements following the CASE statement that has the equivalent value as specified by the SELECT statement. Execution continues until another CASE, CASE ELSE, or END SELECT statement is encountered.

Nesting of the SELECT statement is possible. Internal SELECT contains the entire external SELECT.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.5 BASIC GPIB Control Statement Syntax and Activity

5.5 BASIC GPIB Control Statement Syntax and Activity

This section describes the following statements in order.

1. CLEAR
2. DELIMITER
3. ENTER
4. INTERFACE CLEAR
5. LOCAL
6. LOCAL LOCKOUT
7. OUTPUT
8. REMOTE
9. REQUEST
10. SEND
11. SPOLL
12. TRIGGER

NETWORK ANALYZER
PROGRAMMING MANUAL

5.5 BASIC GPIB Control Statement Syntax and Activity

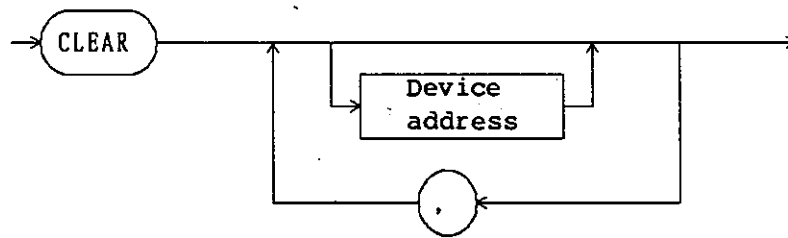
1. CLEAR

Outline

Initialization of all devices, or a specifically selected device connected to the GPIB.

Syntax

(1)-1



(1)-2

CLEAR [Device address {, Device address}]

Commentary

- If only CLEAR is executed without specifying any device address, the universal command DCL is sent to the GPIB. All devices connected to the GPIB are thus initialized.
- If a device address is specified after CLEAR, only the device specified by the device address is addressed, and the address command "select device clear" (SDC) is sent. Hence, only the specified device is initialized. And more than one specific device address can be specified at the same time.

Example

```
10 CLEAR
20 CLEAR 2
30 CLEAR 1 3 5 7
```

Note

CLEAR does not function when in TALKER/LISTENER mode.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.5 BASIC GPIB Control Statement Syntax and Activity

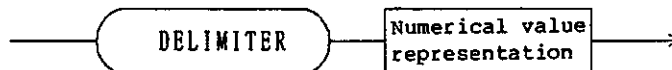
2. DELIMITER

Outline

Statement for selecting and setting one of four delimiters.

Syntax

(1)-1



(1)-2

DELIMITER Numerical value representation

Commentary

- The delimiter corresponding to the number indicated in the numerical value representation is set. The delimiter selection numbers and types are listed below.

Selection	Type of delimiter
0	Output of "CR" and "LF" 2-byte code. Or "LF" output together with "EOI" single wire output.
1	Output of "LF" 1-byte code.
2	Output of "EOI" single wire output together with last byte of data.
3	Output of "CR" and "LF" 2-byte code.

- An error is generated if the Numerical value representation result does lie in the 0 to 3 range, and the value is regarded as an integer with decimal places disregarded.
- "DELIMITER=0" is set automatically when the power is switched on.

Example

```
10 DELIMITER 0  
20 DELIMITER 1  
30 DELIMITER A*10
```

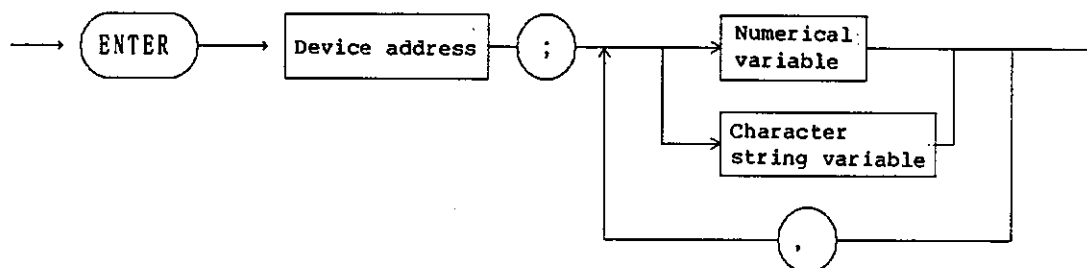
NETWORK ANALYZER
PROGRAMMING MANUAL

5.5 BASIC GPIB Control Statement Syntax and Activity

3. ENTER

Outline Entry of data from GPIB and parallel I/O.

Syntax (1)-1



(1)-2
ENTER Device address; Numerical variable
| Character string variable
{, Numerical variable
| Character string variable }

Note : Device address
0 thru 30 : Address of external device connected to
the GPIB
31 : Data input from the network analyzer
measuring section
34 : Input for F/F state of parallel port
35 : Data input for port C of parallel port
36 : Data input for port D of parallel port
37 : Data input for port CD of parallel port

Commentary

- Input of data via GPIB from device specified by device address, and storage as numerical value or character string within BASIC variable. Note, however, that if the device specified by device address does not have a talker function, the program is stopped without the controller being able to complete the handshake. And if a character string variable is used, that character string must be declared in advance by DIM statement.
- When input is by character string, the length of the character string variable used in the destination must be sufficient to prevent overflow of the input data and disregarding of data which cannot fit in.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.5 BASIC GPIB Control Statement Syntax and Activity

Example

```
10 ENTER 1;A
20 DIM A$(100), B$(20)
30 ENTER 2;A$
40 ENTER 3;B$
```

Note

- Function when in controller mode
Specify the designated address function as TALKER, and accept data.
- Function when in TALKER/LISTENER mode
Time out error is generated if the network analyzer is not specified as TALKER within one minute by external controller.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.5 BASIC GPIB Control Statement Syntax and Activity

4. INTERFACE CLEAR

Outline

Initialization of entire GPIB interface connected to the network analyzer.

Syntax

(1)-1



(1)-2

INTERFACE CLEAR

Commentary

- Execution of this statement results in output of GPIB single wire signal IFC for about 100 microseconds. When the GPIB interface of all devices connected to the network analyzer GPIB receives the IFC signal, the talker or listener status is cancelled.

Example

10 INTERFACE CLEAR

Note

Does not function when in TALKER/LISTENER mode.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.5 BASIC GPIB Control Statement Syntax and Activity

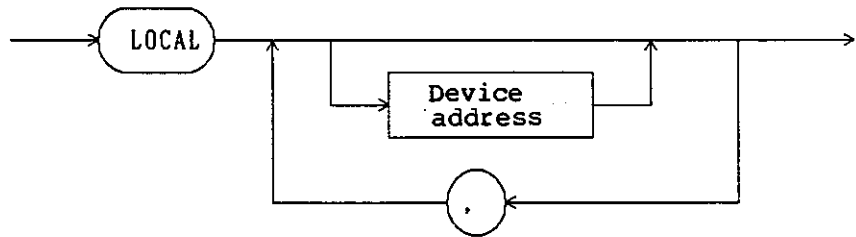
5. LOCAL

Outline

Release of specified device from remote control status, or making the remote enable (REN) line to false.

Syntax

(1)-1



(1)-2

LOCAL [Device address {, Device address }]

Commentary

- If LOCAL is executed without specifying a device address, the GPIB remote control (REN) line becomes false (high level), and all devices connected to the GPIB are switched to local mode. While REN is false, note that GPIB devices cannot be set by OUTPUT command (since GPIB control is no longer effective). To make REN true (low level) again, execute the REMOTE statement.
- If a device address is specified after LOCAL, remote mode can be canceled by addressing only the device specified by that device address.

Example

```
10 LOCAL
20 LOCAL 1
30 LOCAL 1, 2, 3
40 LOCAL A*10+J
```

Note

Does not function when in TALKER/LISTENER mode.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.5 BASIC GPIB Control Statement Syntax and Activity

6. LOCAL LOCKOUT

Outline

Cancellation of the function which enables devices connected to the GPIB to be switched to local mode by front panel operation.

Syntax

(1)-1



(1)-2

LOCAL LOCKOUT

Commentary

- When each device connected to the GPIB is in remote mode (that is, when controlled by remote control via the GPIB) the panel keys on each device are locked to prevent local setting of data. The LOCAL key, remains effective, however, and if pressed, the respective devices are returned to local mode where local setting of data is possible. Consequently, various interruptions during remote control operations are possible, and accurate control may not be possible. By executing the LOCAL LOCKOUT statement, however, the LOCAL key on all devices connected to GPIB can be locked to prevent all local control operations at each device.
- When the LOCAL LOCKOUT statement is executed, the universal command "local lockout" (LLO) is sent to the GPIB.

Example

10 LOCAL LOCKOUT

Note

Does not function when in TALKER/LISTENER mode.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.5 BASIC GPIB Control Statement Syntax and Activity

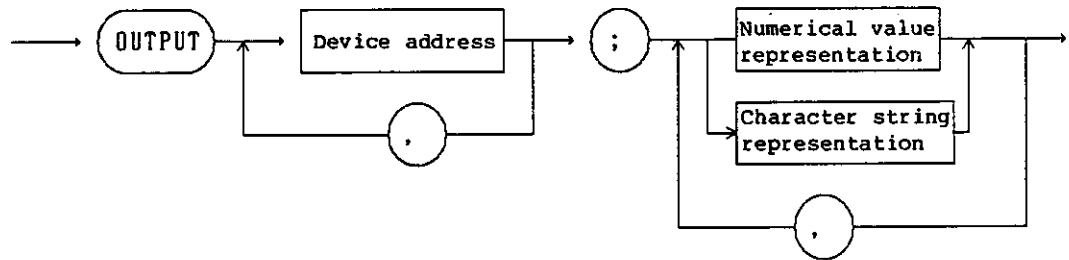
7. OUTPUT

Outline

Output of data to GPIB.

Syntax

(1)-1



(1)-2

```
OUTPUT Device address { , Device address }
; Numerical value representation
| Character sting representation
{ , Numerical value representation
| Character sting representation }
```

Notes : Device address

- 0 thru 30 : Address of external device connected to the GPIB
- 31 : Output to the network analyzer measuring section
- 33 : Output to port A of parallel port
- 34 : Output to port B of parallel port
- 35 : Output to port C of parallel port and F/F set or reset
- 36 : Output to port D of parallel port and port mode set
- 37 : Output to CD port of parallel port

Commentary

- Numerical and character string data is sent as ASCII data to the device specified by device address. More than one device can be specified at once by partitioning device addresses with commas, and numerical value representation and character string representation can even be mixed by also partitioning with commas.
- If the OUTPUT statement is executed when the REN line is true (low level), devices specified by device address are automatically set to remote mode. Remote mode can be cancelled by executing the LOCAL statement.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.5 BASIC GPIB Control Statement Syntax and Activity

Example

```
10 A=5
20 B=10
30 OUTPUT A; "STARTF", B, "MHz"
```

Note

- When in controller mode
Specify the designated address function as LISTENER,
and output data.
- When in TALKER/LISTENER mode
Time out error is generated if the network analyzer is
not specified as LISTENER within one minute by external
controller.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.5 BASIC GPIB Control Statement Syntax and Activity

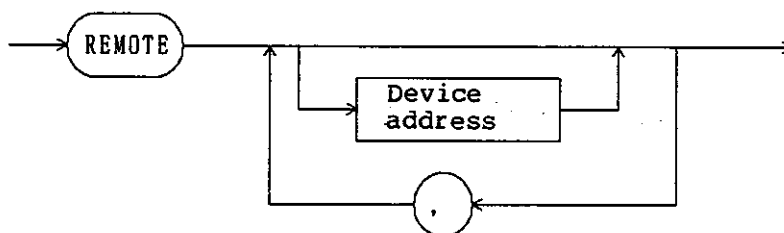
8. REMOTE

Outline

Set specified device to remote mode, or make the GPIB remote enable (REN) line true.

Syntax

(1)-1



(1)-2

REMOTE [Device address {, Device address}]

Commentary

- If only REMOTE is executed without specifying a device address, the GPIB remote enable (REN) line becomes true (low level) and remote control of the devices connected to GPIB becomes possible. The REN line can be made false (high level) by executing the LOCAL statement.
- If a device address is specified after REMOTE, the corresponding device is put into remote mode (as long as the REN line is true (low level)). More than one device address can be specified together. And remote mode can be canceled by executing the LOCAL statement.
- Although the purpose of the REMOTE statement is to put selected devices into remote mode, specified devices are automatically set to remote mode (without executing the REMOTE statement) when any of the following statements is executed (but only as long as the REN line is true (low level)).

```
CLEAR [Device address {, Device address}]  
OUTPUT Device address {, Device address}:  
<output data>{, output data}]  
REMOTE [Device address{,Device address}]  
SEND LISTEN Device address{,Device address}  
TRIGGER Device address{,Device address}
```

Example

```
10 REMOTE 1  
20 REMOTE 5  
30 REMOTE 1, 2, 3, 4  
40 REMOTE A*100+I
```

Note

Does not function when in TALKER/LISTENER mode.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.5 BASIC GPIB Control Statement Syntax and Activity

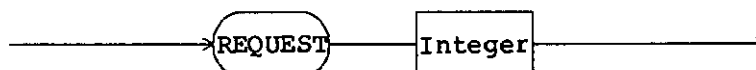
9. REQUEST

Outline

Set status byte to be sent to external GPIB when in TALKER/LISTENER mode.

Syntax

(1)-1



(1)-2
REQUEST Integer

Note : Integer value: 0 thru 255

Commentary

- Set status byte to be sent to external GPIB when in TALKER/LISTENER mode.
- Set a value greater than 64 when generating SRQ.

Example

10 REQUEST 65

Note

Does not function when in controller mode.

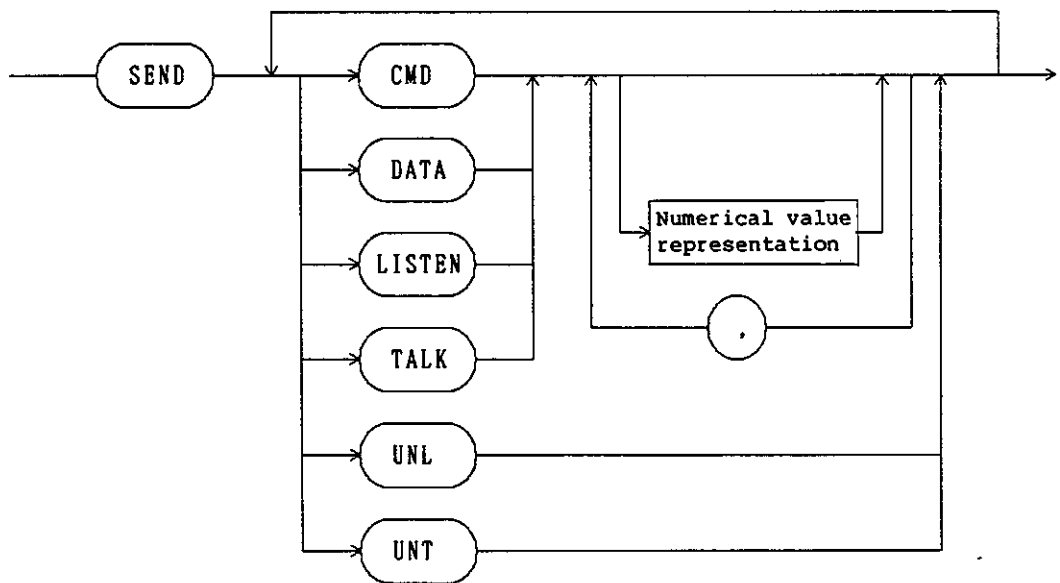
NETWORK ANALYZER
PROGRAMMING MANUAL

5.5 BASIC GPIB Control Statement Syntax and Activity

10. SEND

Outline Output of command and data to GPIB.

Syntax (1)-1



(1)-2
SEND <A>| {,<A> | }

Note : A : <CMD | DATA | LISTEN | TALK >
 [Numerical value representation
 {, Numerical value representation }]
 B : UNL | UNT

Commentary

- Statement for sending universal commands, address commands, and data independently to the GPIB.
- CMD : Make the attention (ATN) line true (low level), and send the given numerical values to the GPIB. Since the numerical values are converted to 8-bit binary data and output to the GPIB, the numbers handled must not exceed the 0 thru 255 range. And numerical values expressed as decimal numbers are automatically converted to integer numbers.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.5 BASIC GPIB Control Statement Syntax and Activity

- DATA** : Make the ANT line false (high level) and sent the given numerical values to the GPIB. The numerical values handled here are subject to the same restrictions as those handled by "CMD".
- LISTEN** : Send the given numerical values to the GPIB as listener address group (LAG). Multiple numbers can also be specified.
- TALK** : Send the given numerical values to the GPIB as talker address group (TAG). Multiple numbers can also be specified.
- UNT** : Send the untalk (UNT) command to the GPIB. Talker mode of the device specified as talker before this command was executed is canceled.
- UNL** : Send the unlisten (UNL) command to the GPIB. Listener mode of the device specified as listener before this command was executed is canceled.

Example

```
10 SEND UNT UNL LISTEN 1, 2, 3 TALK 4
20 SEND UNT CMD 10, 200 DATA 30, 54
```

Note

Does not function when in TALKER/LISTENER mode.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.5 BASIC GPIB Control Statement Syntax and Activity

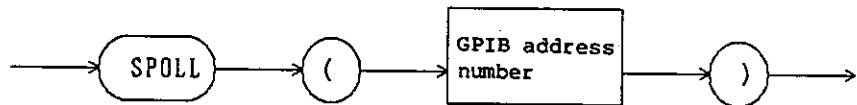
11. SPOLL

Outline

This statement executes serial polling of the specified GPIB equipment to read a status byte.

Syntax

(1)-1



(1)-2

SPOLL (GPIB address number)

Commentary

- The statement executes serial polling of the other GPIB equipment when the network analyzer is in the controller mode.
- The statement executes serial polling of equipment corresponding to each address when the equipment address is 0 to 30.
- The statement takes out a status byte for the network analyzer regardless of the mode, such as controller mode and TALKER/LISTENER mode, when the equipment address is 31.

Example

```
10 ON ISRQ GOSUB 100
20 ON SRQ GOSUB 200
30 ENABLE INTR
40 !
50 GOTO 40
100 S=SPOLL (31)
110 PRONT S
120 RETURN
200 S=SPOLL (1)
210 PRONT S
220 RETURN
```

Note

0 is returned when the equipment address 0 to 30 is specified in the TALKER/LISTENER mode and SPOLL is executed.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.5 BASIC GPIB Control Statement Syntax and Activity

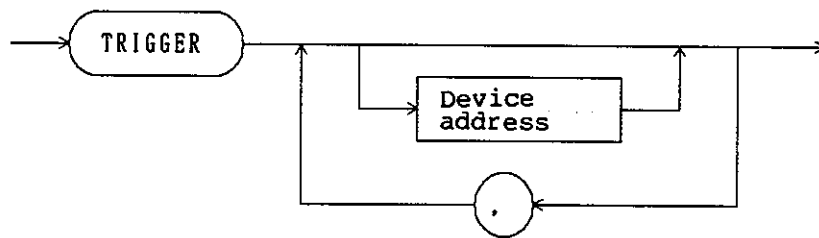
12. TRIGGER

Outline

Send the address command group (ACG) group execute trigger (GET) to all devices, or specifically selected devices connected to the GPIB.

Syntax

(1)-1



(1)-2

TRIGGER [Device address {, Device address }]

Commentary

- If TRIGGER alone is executed without specifying a device address, only the address command "group execute trigger" (GET) is sent to the GPIB. In this case, devices where a trigger is to be applied must be set to listener in advance.
- If a device address is specified after TRIGGER, the GET command is only sent to the specified device.

Example

```
10 TRIGGER 1
20 TRIGGER
30 TRIGGER 2, A*100-J, 30
```

Note

Does not function when in TALKER/LISTENER mode.

5.6 Syntax of BASIC File Control Statement

This section describes the following statements in order.

1. CLOSE
2. COPYFILES
3. DSTAT
4. ENTER
5. ENTER USING
6. OFF END
7. ON END
8. OPEN
9. OUTPUT
10. OUTPUT USING

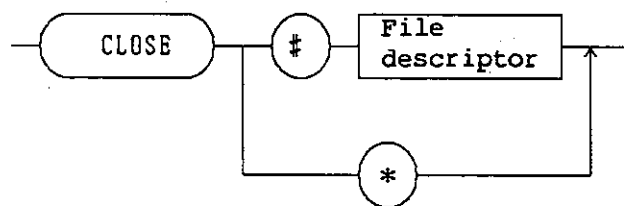
1. CLOSE

Outline

This statement closes the file assigned to the file descriptor.

Syntax

(1)-1



(1)-2

CLOSE #File descriptor |*

Commentary

File opened by the OPEN command must be closed before a floppy disk is removed or the power to the equipment is turned off. Otherwise, data in a file opened for writing is destroyed.

A file is not closed automatically when the BASIC program is stopped by the PAUSE or STOP key. All files are closed when the program ends when it is stopped by a key other than the above. A file is closed when the program ends in error. If the ON ERROR is set, a file is not closed for the erroneous end.

Execute the following close operation explicitly when the program ends in error:

CLOSE *

The above is a specification method to close all files by executing a command.

A file is closed automatically when the SCRATCH or LOAD is executed.

2. COPYFILES

Outline

This statement copies all files in the floppy disk to the other floppy disk by one command.

Syntax

(1)-1



(1)-2

COPYFILES

Commentary

The statement copies all files in a floppy disk to the other floppy disk. Since the system is provided with only one floppy disk, the actual operation needs the following operation in addition to execution of the above command.

Operation to change media is needed. Operation instructions are displayed on the CRT in sequence when the COPYFILES command is specified. Follow these instructions, and the processing will be completed.

The processing is as follows:

- ① Obtain a file name to be copied and size from the directory by executing the command.
- ② Check that the BASIC buffer is provided with an empty area for the above file size.
- ③ If the BASIC buffer is provided with an empty area, read a file to the buffer. Continue this operation until no empty area is found in the buffer or no file to be copied is found in the floppy disk to be copied (source). If the buffer is provided with no empty area, a request to insert a floppy disk to copy (target) is made.
- ④ Set a target floppy disk and press the **Y** and **RETURN** keys.
- ⑤ Output all files copied to the BASIC buffer to the target floppy disk.
- ⑥ If any file is left in the source file, inserting the source floppy disk is requested and the processing is repeated from item 1.

If the capacity of all files to be copied does not exceed the BASIC buffer size, copying can be completed only by inserting a source floppy disk, then a target floppy disk once. When a large number of files is copied, the above cycle must be repeated several times until copying of all the files is completed.

Note

Care must be taken not to insert the source and target floppy disks inversely during copying. Avoid removing the floppy disk during read/write. The temporary storage area uses a different buffer than that used for executing the BASIC program. Executing SCRATCH for the programs in the buffer is recommended to reduce the number of new floppy disks inserted. The reason is that the COPYFILES cannot use the buffer used for these programs. This command is used to copy data when one file is completely stored in BASIC buffer. Unless the file is stored, data is not copied and the command is ignored.

Note that the **STOP** key is not effective during the COPYFILES operation.

3. DSTAT

Outline

This statement inserts data from the directory to the BASIC variable.

Syntax

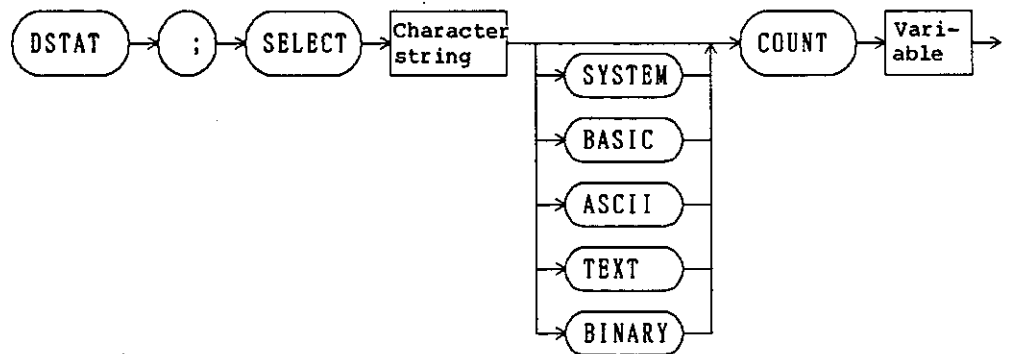
(1)

DSTAT <index><numeric variable>
<index>: 0

(2)

DSTAT
<index><filename><filetype><size><sectors><year><month>
<day><week><hour><minute><start-sector><index>: 1..200

(3)-1



(3)-2

DSTAT; SELECT character string [File type]
COUNT variable

Note : File type: SYSTEM | BASIC | ASCII | TEST | BINARY

Commentary

Syntax (1) is for a command to check the number of files catalogued in the file system directory. The <index> specifies an expression resulting in 0. The second parameter specified is a numerical variable. The execution result is substituted for a numerical variable.

Syntax (2) is for a command to enter the file system directory information to the BASIC variable. The first index specifies an index in the directory by an expression. Values which can be obtained by Syntax 0 to Syntax 1 are available.

The file name specifies a character string variable. Since a file name uses no more than sixteen characters, the length need not be declared.

The third and later parameters specify numerical variables. The following data is substituted:

filetype	File type
	1 BASIC
	2 SYSTEM
	3 ASCII
	4 TEXT
	5 BINARY
	6 DATA
size	File size (the number of bytes)
sectors	Number of sectors
year,month,day	File creation year and date
week	1988 is assumed to be 1.
hour, minute	Sunday is assumed to be 0.

Variable specification can be omitted for an unnecessary value. File name and creation year and date can be obtained as follows.

```
DSTAT | FNAME$,,,,year,month,day
```

The above syntax is substituted in a variable to specify the number of the following files after the COUNT: files of character strings specified by the SELECT and files whose file types are specified in numerical expressions.

Example

```
DSTAT ; SELECT "FILE", COUNT NUM
```

SELECT

This statement searches a character string after the SELECT from the disk as a file name. When a character string includes the following characters (metacharacters), that character string has a special meaning. The following characters used in a file name are also assumed to be metacharacters:

- ? : Matches one character.
- * : Matches one or more characters.
- [] : Matches a character in a character string surrounded with brackets, []. Matches a character in a range from the first character to the second character by specification of [character - character].

NETWORK ANALYZER
PROGRAMMING MANUAL

5.6 Syntax of BASIC
File Control Statement

DSTAT ; SELECT"PROG?.*",COUNT A

The file type specifies one of SYSTEM, BASIC, ASCII, TEXT, or BINARY. A file of the specified file type is searched from the floppy disk.

COUNT

This statement substitutes the number of the file searched by the SELECT for a variable.

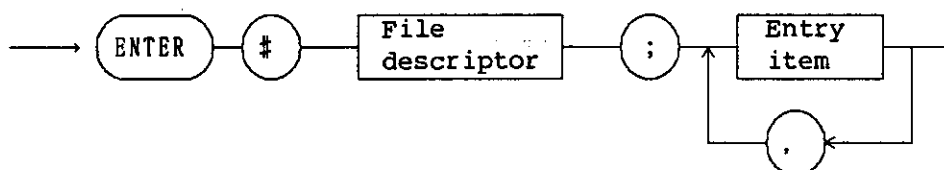
4. ENTER

Outline

This statement reads data from the file and substitutes it for the entry item.

Syntax

(1)-1



(1)-2

ENTER #File descriptor ; Entry item { , Entry item }

Commentary

The statement reads data in the data type format of the corresponding entry item from the file assigned to the file descriptor, and substitutes it for the entry item.

Example

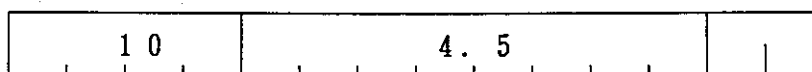
- ① BINARY file
The BINARY file expresses the internal data without change.
The BINARY file reads four bytes of header when an entry item is an integer or character string, or eight bytes of header when an entry file is a real number. Then, it reads the data for the length specified by the header.
Since the number or bytes to be read depends on the entry item type, correct data cannot be obtained unless the same entry type as for output is entered.

```
10 INTEGER I  
20 DIM R  
30 OPEN "FILE" FOR INPUT AS #FD  
40 ENTER #FD; ,I,R,S$
```

NETWORK ANALYZER
PROGRAMMING MANUAL

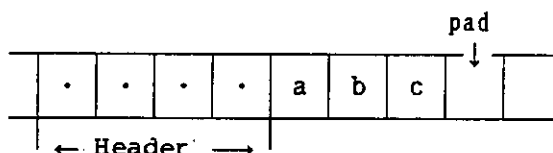
5.6 Syntax of BASIC
File Control Statement

Number of bytes to be read depends on the entered variable type.



When a variable is a real number, eight bytes of data are read and substituted in the variable without change.

When a variable is an integer, four bytes of data are read and substituted in the variable without change.

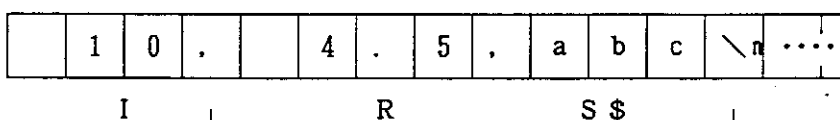


When a variable is a character string, four bytes of header and the data for the length specified by the header are read, and substituted in the character string.

② TEXT file

The TEXT file reads up to line feed regardless of the number of entry items. Data up to comma (,) is assumed to be one item of data, converted to the corresponding entry item type, and substituted. When the number of entry items are larger than the actual data, the last stored data item remains in the excessive variables. Inversely, when the number of variables is smaller than the actual data, the excess data is discarded.

```
10 INTEGER I
20 DIM R
30 OPEN "FILE" FOR INPUT AS #FD;TEST
40 ENTER #FD;I,R,S$
```



Line feed is provided at the end of the item.

Each item is divided by a comma.

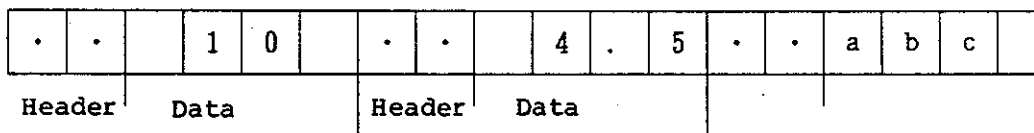
NETWORK ANALYZER
PROGRAMMING MANUAL

5.6 Syntax of BASIC
File Control Statement

③ ASCII file

The ASCII file reads two bytes of header and the data for the length specified by the header. It converts the data according to the variable type, and substitutes it for the variable.

```
10 INTEGER I
20 DIM R
30 OPEN "FILE" FOR INPUT AS #FD;ASCII
40 ENTER #FD;I,R,S$
```



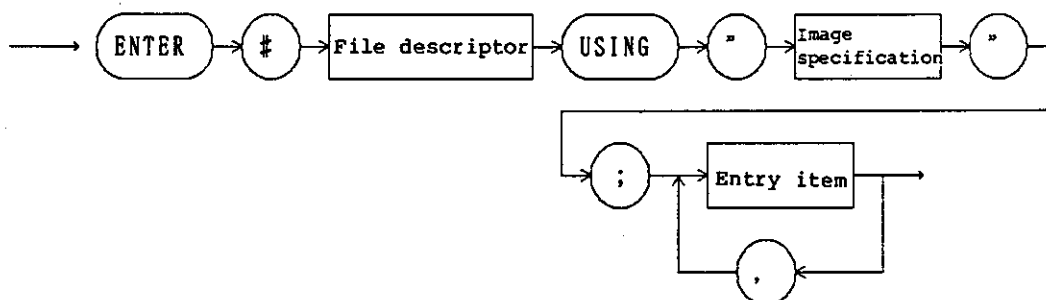
5. ENTER (ENT) USING (USE)

Outline

This statement enters data from the file to an entry item in the image specifications format.

Syntax

(1)-1



(1)-2

ENTER #File descriptor USING "image specifications"
;Entry item {,Entry item}

Commentary

Entry item Entry item statement enters data from the file assigned to the file descriptor to an entry item in the image specifications format.

Image specifications

- D : A value is read assuming that the number of Ds is the number of digits of that value, and substituted for a variable of an entry item.
- Z : The same as D.
- K : One line is read, converted to numerical data, and substituted for a variable for an entry item.
- S : The same as D.
- M : The same as D.
- . : The same as D.
- E : The same as K.
- H : The same as K, but the value is converted to the European numerical format (a comma is used as decimal point).
- * : The same as D.
- A : Characters are read for the number of As and substituted for the character string variable.
- k : One line is read and substituted for a character string variable.
- X : One character data is skipped.
- Literal : A character string closed by \" is skipped.

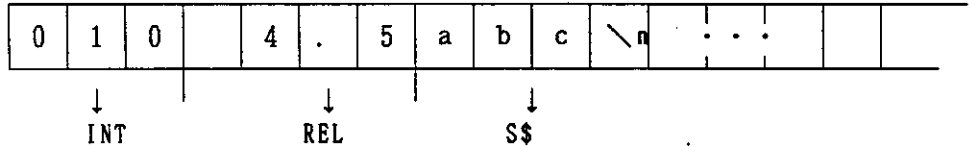
NETWORK ANALYZER
PROGRAMMING MANUAL

5.6 Syntax of BASIC
File Control Statement

B : A character is read and substituted for an
entry item as an ASCII code.
@ : One byte data is skipped.
+ : The same as the @.
- : The same as the @.
: It is ignored by the ENTER statement.
n : Repeat of image specifications can be
specified with the number. 3D.2D is the same
as DDD.DD. 4A is the same as AAAA.

Example

```
10 INTEGER INT
20 DIM REL
30 ENTER $FD USING "ZZZ,DD.D,3A";INT,REL,S$
```

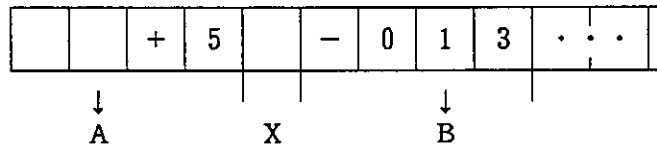


INT : Three bytes of data are read, converted to the integer type of the INT data type and substituted for the INT. The INT value is set to 10 after execution.

REL : 'DD.D' of the image specifications corresponds to the REL of an entry item. Four bytes data are read, converted to the real number type, and substituted for the REL. The REL is set to 4.5 after execution.

S\$: Three bytes of data are read and substituted for S\$. The S\$ is set to abc after execution.

```
10 DIM A,B
20 ENTER #FD USING "SDDD,X,MZZZ";A,B
```



A,B : Four bytes of data are read, converted to the real number type, and substitutes for A and B. A and B are set to 5.0 and 13.0 after execution. One byte for X of the image specifications is read, but no data is substituted for a variable. Data entered in the SDDD format is read and substituted for A. X does not need a variable, and one character is skipped. Four bytes are entered, converted to the real number type, and substituted for B assuming that 'MZZZZ' corresponds to B.

```
10 DIM A
20 ENTER #FD USING "K";a
```

S	T	R	I	N	G	1	2	3	.	5	#	#	\n	.	.
---	---	---	---	---	---	---	---	---	---	---	---	---	----	---	---

A is set to 123.5 after execution.

'STRING123.5##' is read and converted to the real number type of the entry variable A.

When an entry item is the real number type, characters other than preceding values, codes (+,-), and indexes E and e are ignored, and only numerals are accepted.

Conversion to numerals stops at the position where a character other than numerical is encountered.

Since line feed is used as a terminator for K, E, k, and H of the image specifications, the data is substituted to a variable assuming that the data from the current file pointer to line feed is one item.

6. OFF END

Outline

This statement clears the processing for the end of file specified in the ON END statement.

Syntax

(1)-1



(1)-2

OFF END #File descriptor

Commentary

When the end of file occurs after the destination of the branch defined in the file descriptor is cleared, the error message below is displayed and the system control ends.

end of "DATAFILE" file

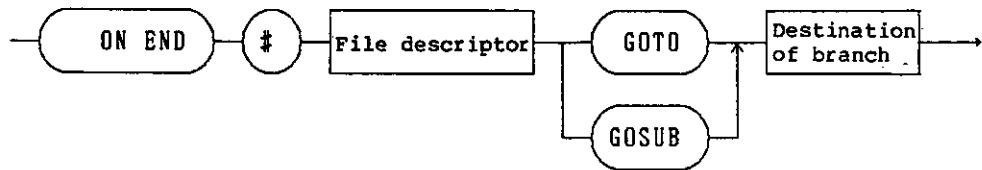
7. ON END

Outline

This statement defines the processing (destination of branch) for the end of file.

Syntax

(1)-1



(1)-2

ON END #File descriptor GOTO | GOSUB Destination of branch

Commentary

End of file occurs when data is read from the file by the ENTER statement until the end of file is reached and no data to be entered is found. The error message is displayed and the system control ends after the file is closed unless the processing is declared by the ON END statement.

Destination of the branch is specified in a numerical variable, numerical constant, or label.

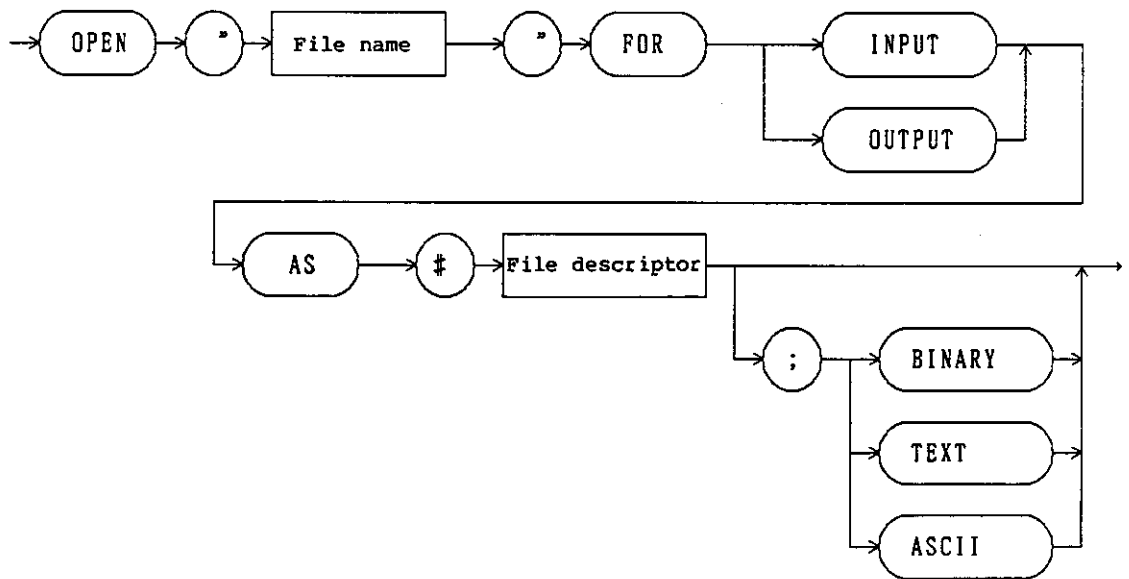
8. OPEN

Outline

This statement assigns the file descriptor to the file and opens it in the specified processing mode.

Syntax

(1)-1



(1)-2

OPEN_"file.name" FOR Processing mode AS #File descriptor
[;Type]

Note : Processing mode : INPUT | OUTPUT

 Type : BINARY | TEXT | ASCII

Commentary

The statement assigns the file descriptor to the file to make the program recognize the file and opens it in the specified processing mode.

Processing mode

Processing mode has two types : OUTPUT and INPUT.
OUTPUT is used to write the file II data and INPUT used to read data from the file.

#File descriptor

ENTER/OUTPUT is used to write/read an actual file. The file descriptor is used to make these commands recognize the file to be processed.
The file descriptor name is described by alphanumeric after #.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.6 Syntax of BASIC
File Control Statement

File type

File type consists of BINARY, TEXT, and ASCII.
If no file type is specified, BINARY is assumed.

BINARY is used to record data with internal expression. Four bytes or eight bytes are recorded if the data is an integer or real number. Four bytes of header are followed by ASCII data if the data is a character string. Space for one byte is provided after the data if the number of data characters is an odd number.

TEXT is used to convert data to ASCII code and output. "-" or a space is provided before a value.
USING can be specified in the TEXT file.

ASCII is used to express entry and output items with ASCII codes after two bytes header. "-" or a space is provided before a value. One byte of space is provided after the data if the number of data characters is an odd number.

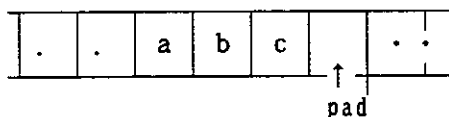
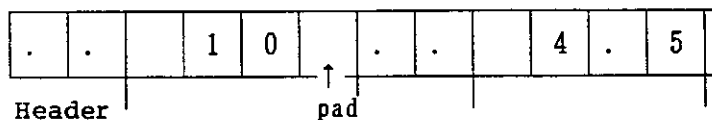
- When the file descriptor assigned to the other file is opened, the last assigned file is closed and the newly specified file is opened.
- The same file cannot be opened at the same point by multiple file descriptors.
- If an existing file is opened in the OUTPUT mode, an error message is displayed and the program stops. This operation avoids deleting a necessary file erroneously. To create a new file whose name is the same as that of an existing one, delete an existing file by the PURGE command.

Example

```
10 OPEN "DATA.BAS" FOR OUTPUT AS #fd ; TEXT
20 OUTPUT #FD;10,4.5"abc"
```



```
10 OPEN "DATA.BAS" FOR OUTPUT AS #FD ; ASCII
20 OUTPUT #FD;10,4.5,"abc"
```



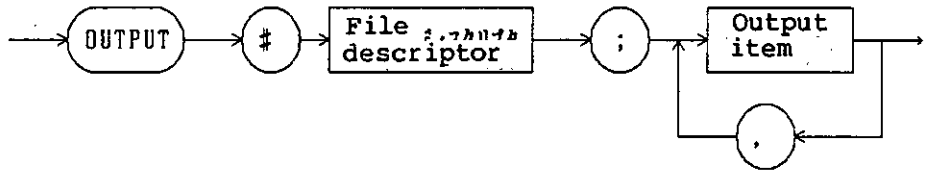
9. OUTPUT (OUT)

Outline

This statement outputs (or writes) the data assigned to the #file descriptor.

Syntax

(1)-1



(1)-2

OUTPUT #File descriptor ; Output item { , Output item }

Commentary

The statement converts output items to the BASIC standard format to be output.

The file descriptor specified when the file is opened is used. The file descriptor is assigned to the file to be processed when the file is opened. The subsequent processing for the file is always performed via this file descriptor.

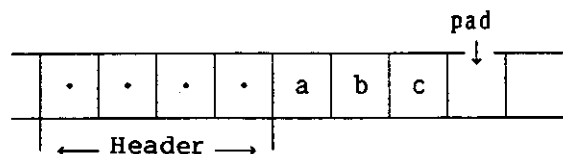
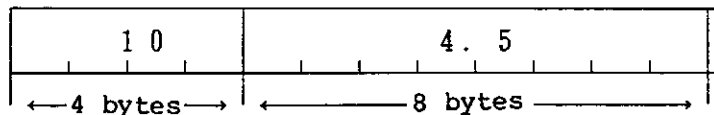
Example

①

BINARY file

The data is output is the same type as the internal expression. The character string is output with a four-byte header indicating the length of the character string. When the number of characters of the character string is an odd number, a space for one character is provided at the end of the characters.

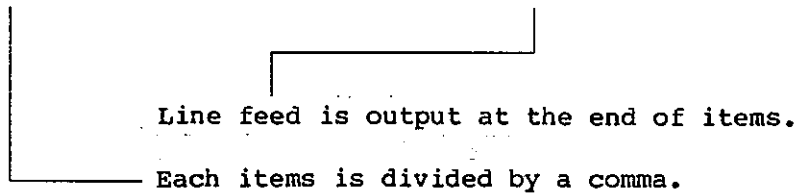
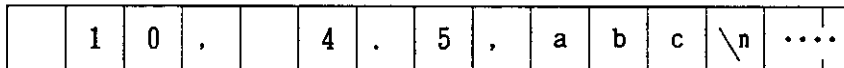
```
10 OPEN "FILE" FOR OUTPUT AS #FD
20 OUTPUT #FD; 10,4.5,"abc"
```



The length of the header is the same as the data.

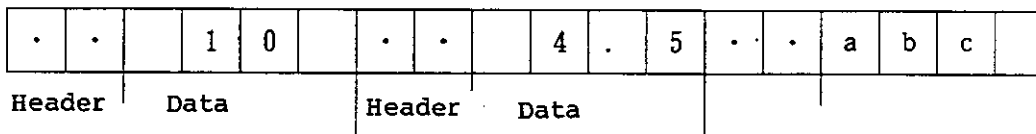
- ② TEXT file
Data is converted to ASCII code and output. "-" or a space is followed by numerical data.

```
10 OPEN "FILE" FOR OUTPUT AS #FD ;TEXT
20 OUTPUT #FD;d 10,4.5,"abc"
```



- ③ ASCII file
Data is converted to ASCII code and output. "-" or a space is followed by numerical data, a Space is provided at the end of the data when the number of bytes of data is an odd number.

```
10 OPEN "FILE" FOR OUTPUT AS #FD ;ASCII
20 OUTPUT #FD; 10,4.5,"abc"
```



The length of the header is the same as that of the data.

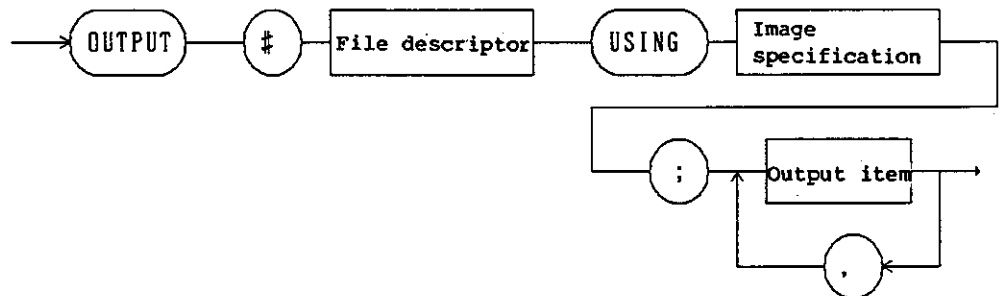
10. OUTPUT (OUT) USING

Outline

This statement output (writes) data to the file assigned to the \$file descriptor in the specified format.

Syntax

(1)-1



(1)-2

OUTPUT #File descriptor USING image specifications
;Output item { ,Output item }

Commentary

The statement converts the format freely to output data by specifying the USING and the image specifications. The image specifications are specified in the character string format.

The file descriptor specified when the file is opened is used. The file descriptor is assigned to the file to be processed when the file is opened. The subsequent processing for the file is always performed via this file descriptor.

Image specifications

- D : The number of digits to output a value is specified by the number of Ds. A blank in the specified field is provided by a space.
- Z : The number of digits to output a value is specified by the number of Zs. A blank in the specified field is provided by a 0.
- K : The expression value is output in the BASIC standard format (the same as the PRINT).
- S : Plus (+) or minus (-) is output to the S position.
- M : Minus (-) for a negative value or a space for a positive value is output to the M position.
- . : Alignment is done so that a decimal point is on the position ".".
- E : Outputs the format e code exponent.

NETWORK ANALYZER
PROGRAMMING MANUAL

5.6 Syntax of BASIC
File Control Statement

- H : The same as K, but a comma is used as a decimal point.
- R : The same as ".", but a comma is used as a decimal point.
- * : The number of digits to output a value is specified by the number of asterisks (*). * is output to a blank in the specified field.
- A : One character is output to the position A.
- k : The value of a character string is output without change.
- Literal : A character string closed by "\" is output without change regardless of the output item.
- X : A space is provided for the X position.
- B : An expression value is accepted as an ASCII code.
- @ : Form feed is output.
- + : Carriage return is output.
- : Line feed is output.
- # : Line feed is provided at the end of items automatically. Line feed is not provided if this image is specified.
- n : The number of repetitions of each image specification is specified by a numeral. For example, 3D.2D means DDD.DD and 4A means AAAA.

Example

OUTPUT #FD USING "ZZZ,DD.D,3A";10;4.5;"abc"

0	1	0		4	.	5	a	b	c	\n	...
---	---	---	--	---	---	---	---	---	---	----	-----

↑ "abc" is converted to the format of image specification "3A" and is output.

↑ 4.5 is output in the format: 出力します. of "DD. D".

↑ 10 is output in the format of "ZZZ".

OUTPUT #FD USING "SDDD,X,MZZZ";+5,-13.57

		+	5		-	0	1	4	.	.	.
--	--	---	---	--	---	---	---	---	---	---	---

↑ The first decimal place of 13.57 is rounded off. Three digits of integers are entered.

↑ A space for one byte is provided.

↑ A four-byte area is provided and is output with a code.

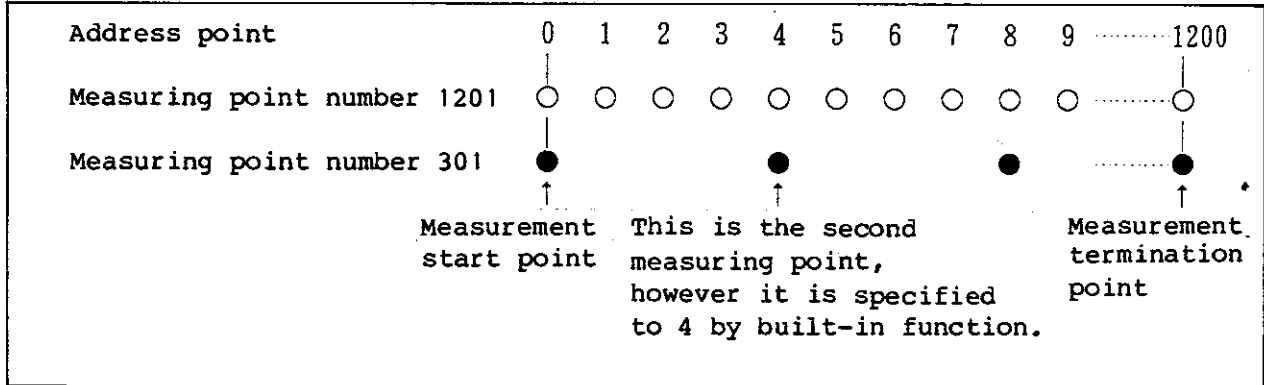
6. BUILT-IN FUNCTIONS

6.1 Outline

Built-in functions are functions incorporated in the network analyzer for use in CPU high-speed calculations and evaluations of various different operations ranging from analysis of input data to GO and NG judgments. Since the 64-bit high-speed operations executed internally, do not require the wasteful data transfers common in more conventional chips, processing efficiency has been greatly improved.

(1) Measuring Point and Address Point

This section describes the measuring point and address point which are sometimes misused to operate the built-in function. The network analyzer selects several points in the frequency range to measure. The number of points measured actually is called measuring point number. The built-in function specifies the measuring point at the address point regardless of measuring point number. The address point ranges from 0 to 1200.



(2) Response to Built-in Function

Response to built-in function can be classified into three types.

Table 6 - 1 Type of Response to Built-in Function

Response type	Description
meas.point	Function for measuring point only
1201 point	Function for all 1201 points Function other than the measuring point is an approximation to the straight line.
compensate	Function returns an approximation to the straight line even though no address point is in specified level or frequency.

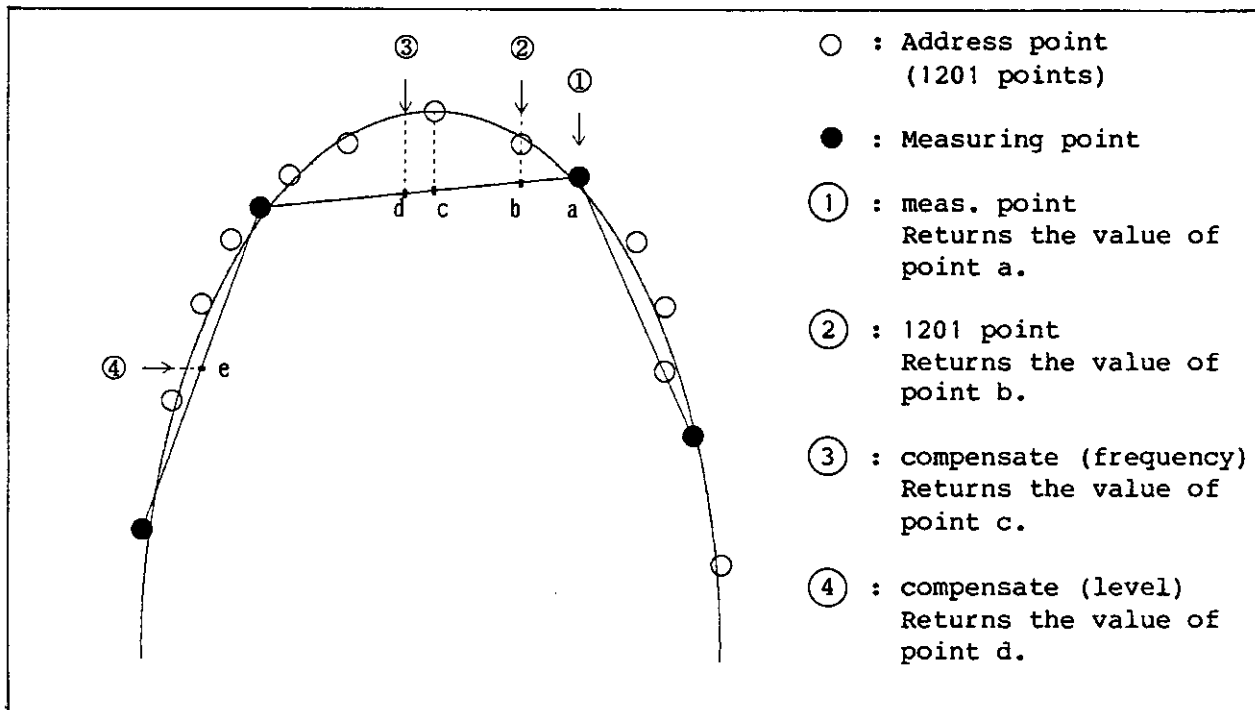


Figure 6 - 1 Details of Response Type

Functions giving the point are listed and shown in Table 6-1 and Figure 6-1, respectively. Functions giving frequency are described below.

① meas. point

POINT1 is the only function that gives frequency. This function returns the number of measurement point approximate to frequency given.

If frequency ② is given in Figure 6-1, the value of point a is returned.

② 1201 point

If frequency like ③ in Figure 6-1 is given, the function for 1201 point converts the frequency to the nearest address point number, then approximates the value corresponding to this point to the straight line.

If frequency ③ is given in Figure 6-1, the value of point c is returned.

③ compensate

If frequency ③ is given in Figure 6-1, the value corresponding to the frequency (value of point d) is approximated to the straight line.

NETWORK ANALYZER
PROGRAMMING MANUAL

6.2 List of Built-in Functions

6.2 List of Built-in Functions

Built-in Function	Response type	Details of output
(1) Frequency Point No.		
POINT1(F, M) POINT2(F, M) DPOINT(F ₀ , F ₁ , M)	meas.point 1201 point 1201 point	Point No. for specified frequency Point No. for specified frequency Difference in point No. between specified frequencies
(2) Point No. Frequency		
FREQ(P, M) DFREQ(P ₀ , P ₁ , M)	1201 point 1201 point	Frequency for specified point Frequency bandwidth between specified points
(3) Point No. Response Value		
VALUE(P, M) DVALUE(P ₀ , P ₁ , M)	meas.point meas.point	Response value for specified point Difference in level between specified points
(4) Frequency Response Value		
CVALUE(F, M) DCVALUE(F ₀ , F ₁ , M)	compensate compensate	Response value for specified frequency Difference in level between specified frequencies
(5) Functions Which Include Search Functions		
① Max Search Function MAX(P ₀ , P ₁ , M) FMAX(P ₀ , P ₁ , M) PMAX(P ₀ , P ₁ , M)	meas.point meas.point meas.point	Maximum response value between specified points Frequency for the maximum response value between specified points Point No. for the maximum response value between specified points
② Min Search Function MIN(P ₀ , P ₁ , M) FMIN(P ₀ , P ₁ , M) PMIN(P ₀ , P ₁ , M)	meas.point meas.point meas.point	Minimum response value between specified points Frequency for the minimum response value between specified points Point No. for the minimum response value between specified points

NETWORK ANALYZER
PROGRAMMING MANUAL

6.2 List of Built-in Functions

Built-in Function	Response type	Details of output
(6) Band Width Calculation Function		
BND(P, X, M)	compensate	Bandwidth from specified point to XdB down
BNDL(P, X, M)	compensate	Low frequency for bandwidth from specified point to XdB down
BNDH(P, X, M)	compensate	High frequency for bandwidth from specified point to XdB down
CBND(F, X, M)	compensate	Bandwidth from specified frequency to XdB down
CBNDL(F, X, M)	compensate	Low frequency for bandwidth from specified frequency to XdB down
CBNDH(F, X, M)	compensate	High frequency for bandwidth from specified frequency to XdB down
(7) Ripple Functions		
① Differential Coefficient DIFFX(ΔX , ΔY , M)	1201 point	Convert ΔX (frequency) to ΔX (point).
DIFFY(ΔX , ΔY , M)	1201 point	Convert ΔY without changing $\Delta Y/\Delta X$.
② Ripple Detection Function (I) RPL1(P ₀ , P ₁ , ΔX , ΔY , M)	1201 point	Difference between the maximum value of maximum point and the minimum value of maximum point.
RPL2(P ₀ , P ₁ , ΔX , ΔY , M)	1201 point	Maximum difference between adjacent maximum and minimum points
RPL3(P ₀ , P ₁ , ΔX , ΔY , M)	1201 point	Maximum internal value to which difference between adjacent maximum and minimum points has been added
③ Ripple Detection Function (II) RPLF(P ₀ , P ₁ , ΔX , ΔY , M)	1201 point	Frequency difference between maximum and minimum points
RPLR(P ₀ , P ₁ , ΔX , ΔY , M)	1201 point	Response value difference between maximum and minimum points

NETWORK ANALYZER
PROGRAMMING MANUAL

6.2 List of Built-in Functions

Built-in Function	Response type	Details of output
④ Maximum, minimum point detection function RPLH(P ₀ , P ₁ , X, Y, M) FRPLH(P ₀ , P ₁ , X, Y, M) PRPLH(P ₀ , P ₁ , X, Y, M) RPLL(P ₀ , P ₁ , X, Y, M) FRPLL(P ₀ , P ₁ , X, Y, M) PRPLL(P ₀ , P ₁ , X, Y, M) NRPLH(P ₀ , P ₁ , X, Y, M) NRPLL(P ₀ , P ₁ , X, Y, M) PRPLHN(N, M) PRPLLN(N, M) FRPLHN(N, M) FRPLLN(N, M) VRPLHN(N, M) VRPLLN(N, M)	1201 point 1201 point 1201 point 1201 point 1201 point 1201 point 1201 point meas.point meas.point meas.point meas.point meas.point	Response value of maximum point Maximum point frequency no. of maximum point Response value of minimum point Minimum point frequency Point no. of minimum point Number of maximum points Number of minimum points Point no. of Nth maximum point Point no. of Nth minimum point Frequency of Nth maximum point Frequency of Nth minimum point Response value of Nth maximum point Response value of Nth minimum point
(8) Other Functions		
① Limit test LMTUL1(X, Up, Lo) LMTUL2(P, Up, Lo, M) LMTMD1(X, Md, D1) LMTMD2(P, Md, D1, M)	1201 point 1201 point 1201 point 1201 point	Return the following values to specified range. Within the range : 0 For more than the upper value : 1 For more than the lower value : 2 For an error : -1
② Zero phase detection function ZEROPHS(P ₀ , P ₁ , M)	meas.point	Frequency of Zero Phase
③ Direct search function DIRECT(P ₀ , P ₁ , X, M) CDIRECT(F ₀ , F ₁ , X, M) DDIRECT(P ₀ , P ₁ , X, M) CDDIRECT(F ₀ , F ₁ , X, M)	1201 point compensate 1201 point compensate	Measuring point no. of the response value Frequency of the response value Difference of measuring points of the response value Frequency difference of the response values

NETWORK ANALYZER
PROGRAMMING MANUAL

6.2 List of Built-in Functions

CAUTION

1. The following functions cannot be used for Log Sweep.
POINT2, DPOINT, CVALUE, DCVALUE, BND, BNDL, BNDH, CBND, CBNDL, CBNDH, ZEROPHS, functions referring to Ripple, CDIRECT and CDDIRECT
2. The following functions cannot be used for Cw Sweep.
POINT2, DPOINT, DFREQ, CVALUE, DCVALUE, BND, BNDL, BNDH, CBND, CBNDL, CBNDH, ZEROPHS, functions referring to Ripple, DIRECT, DDIRECT, CDIRECT and CDDIRECT
3. The following functions cannot be used for Level Sweep.
BND, BNDL, BNDH, CBND, CBNDL, CBNDH, ZEROPHS and functions referring to Ripple
4. The following functions cannot be used during parameter conversion is ON.
BND, BNDL, BNDH, CBND, CBNDL, CBNDH and functions referring to Ripple.

6.3 Description of Built-in Function

Item 6.3 describes built-in functions, however the following notes are considered.

CAUTION

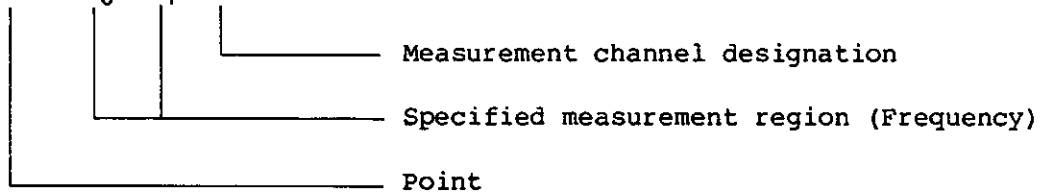
1. Even through $P_0 > P_1$ and $F_0 > F_1$, the function specifying measurement area changes the value to process.
2. If the value exceeding the range of screen setting frequency is specified, the function specifying frequency causes an error.
3. If the value other than 0 to 1200 is specified, the function using the address point causes an error.
4. If an error occurs, the function using the address point returns an error message and -1.
The other function returns an error message and an irregular value.

NETWORK ANALYZER
PROGRAMMING MANUAL

6.3 Description of Built-in Function

DPOINT function : If the frequency width is specified, that frequency width is taken as the measurement point inside the measuring device to calculate the point to which it corresponds.

DPOINT (F₀, F₁, M)



M=0 (1CH)) Data
M=1 (2CH))
M=2 (1CH)) Memory
M=3 (2CH))

(1201 POINT)

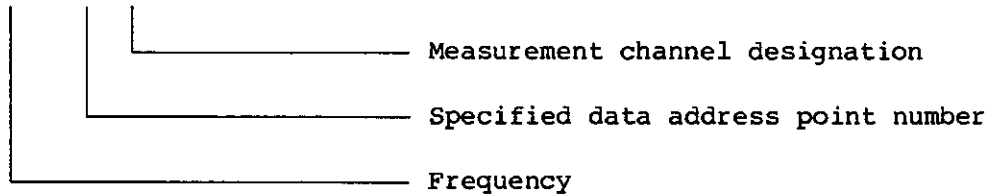
NETWORK ANALYZER
PROGRAMMING MANUAL

6.3 Description of Built-in Function

(2) Function Determining Frequency - FREQ, DFREQ

FREQ function : Calculates frequency corresponding to the point and returns it if the address point is specified.

FREQ (P, M)

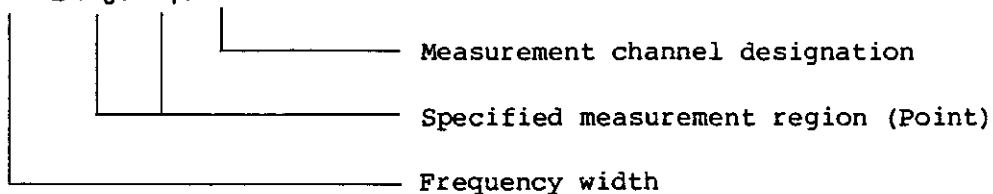


M=0 (1CH)) Data
M=1 (2CH))
M=2 (1CH)) Memory
M=3 (2CH))

(1201 POINT)

DFREQ function : Calculates frequency width corresponding to the width of point by address point and returns it.

DFREQ (P₀, P₁, M)



M=0 (1CH)) Data
M=1 (2CH))
M=2 (1CH)) Memory
M=3 (2CH))

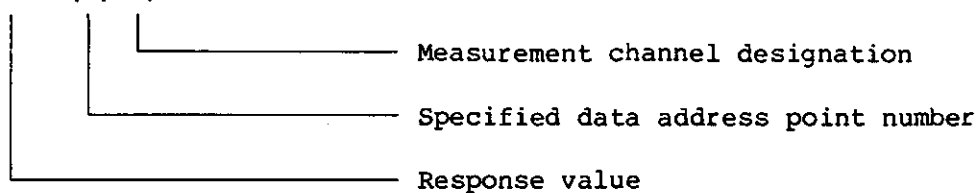
(1201 POINT)

6.3.2 Function Determinating Response Value

(1) Function Determining Response Value from Address Point - VALUE, DVALUE

VALUE function : Returns measured response value at the point if the address point is specified.

VALUE (P, M)

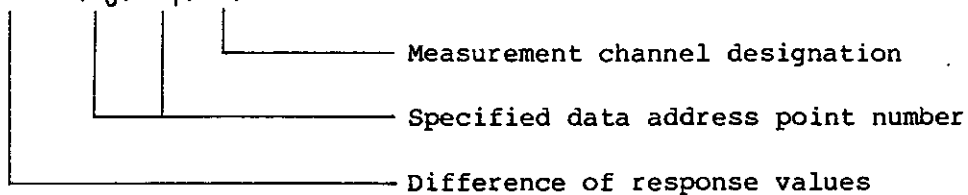


M=0 (1CH) Data
M=1 (2CH)
M=2 (1CH) Memory.
M=3 (2CH)

(MEAS POINT)

DVALUE function : Calculates a difference in measured response value between two points specified by address point and returns it.

DVALUE (P₀, P₁, M)



M=0 (1CH) Data
M=1 (2CH)
M=2 (1CH) Memory
M=3 (2CH)

(MEAS POINT)

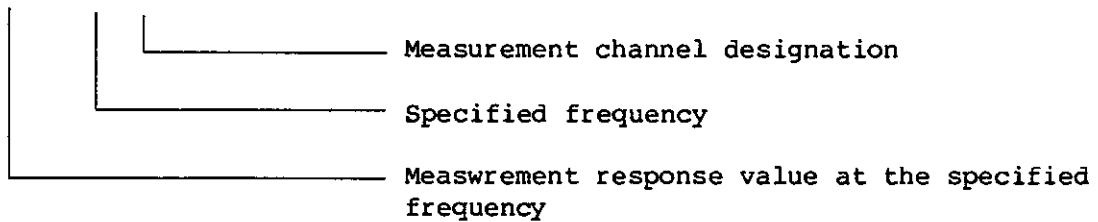
NETWORK ANALYZER
PROGRAMMING MANUAL

6.3 Description of Built-in Function

(2) Function Determining Response Value from Frequency - CVALUE, DCVALUE

CVALUE function : If frequency is specified, the measurement response value at that frequency is displayed.

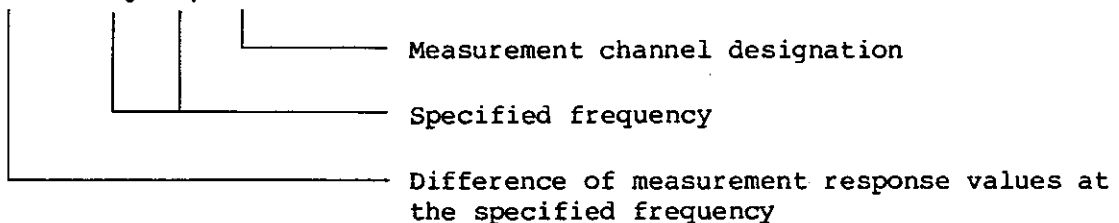
CVALUE (F, M)



M=0 (1CH) Data
M=1 (2CH))
M=2 (1CH) Memory
M=3 (2CH))

DCVALUE function : If two frequencies are specified, the difference between the measurement response values at those frequencies is displayed.

DCVALUE (F₀, F₁, M)



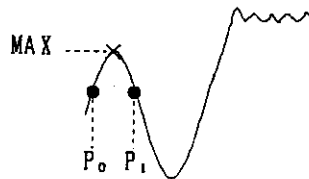
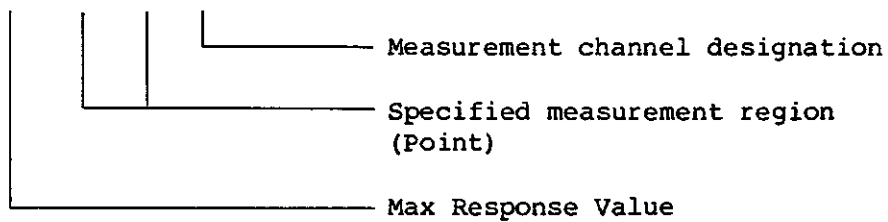
M=0 (1CH) Data
M=1 (2CH))
M=2 (1CH) Memory
M=3 (2CH))

6.3.3 Functions Which Include Search Functions

(1) Function Determining the Maximum Response Value - MAX, FMAX, PMAX

MAX function : If the measurement region is specified by address point, the maximum value in that region is returned.

MAX(P₀, P₁, M)

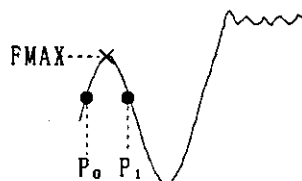
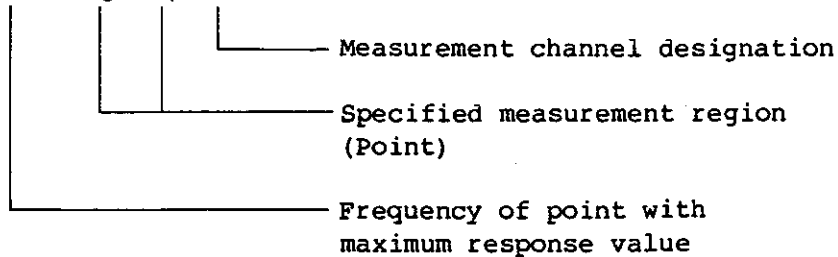


- M=0 (1CH) Data
- M=1 (2CH) Data
- M=2 (1CH) Memory
- M=3 (2CH) Memory

(MEAS POINT)

FMAX function : If the measurement region is specified by address point, the frequency of the point with the maximum response value in that region is returned.

FMAX(P₀, P₁, M)

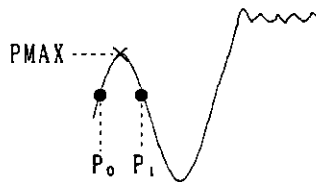
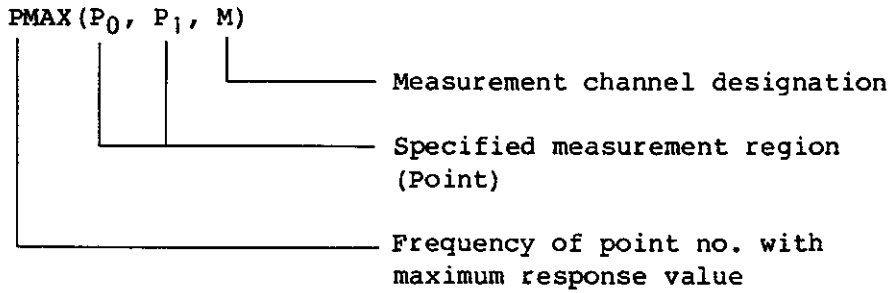


- M=0 (1CH) Data
- M=1 (2CH) Data
- M=2 (1CH) Memory
- M=3 (2CH) Memory

(MEAS POINT)

6.3 Description of Built-in Function

PMAX function : If the measurement region is specified by address point, the point no. with the maximum response value in that region is returned.

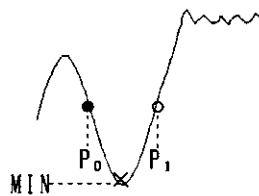
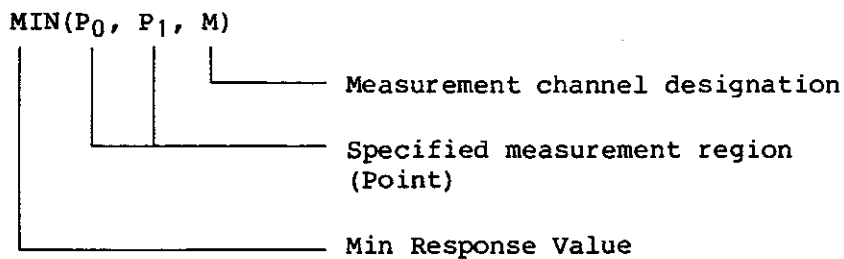


M=0 (1CH) Data
M=1 (2CH) Memory
M=2 (1CH) Memory
M=3 (2CH) Memory

(MEAS POINT)

(2) Function Determining the Minimum Response Value - MIN, FMIN, PMIN

MIN function : If the measurement point region is specified by address point, the minimum response value in that region is returned.



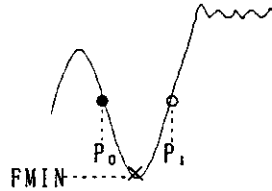
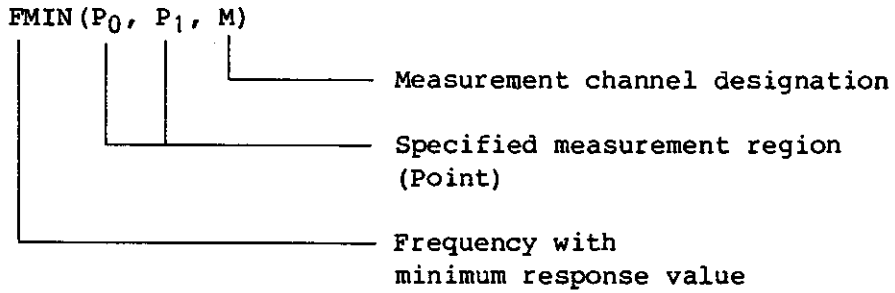
M=0 (1CH) Data
M=1 (2CH) Memory
M=2 (1CH) Memory
M=3 (2CH) Memory

(MEAS POINT)

NETWORK ANALYZER
PROGRAMMING MANUAL

6.3 Description of Built-in Function

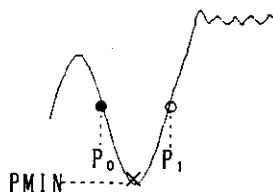
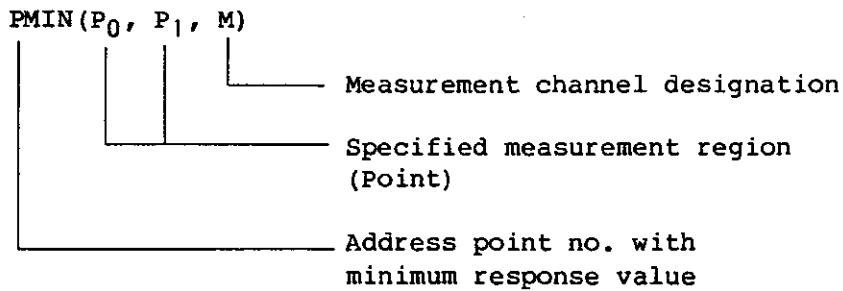
FMIN function : If the measurement point region is specified by address point, the frequency of the point with the minimum response value in that region is returned.



- M=0 (1CH)) Data
- M=1 (2CH))
- M=2 (1CH)) Memory
- M=3 (2CH))

(UNCOMPENSATE)

PMIN function : If the measurement region is specified by address point, the point no. with the minimum response value in that region is returned.



- M=0 (1CH)) Data
- M=1 (2CH))
- M=2 (1CH)) Memory
- M=3 (2CH))

(MEAS POINT)

NETWORK ANALYZER
PROGRAMMING MANUAL

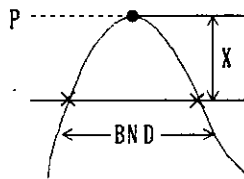
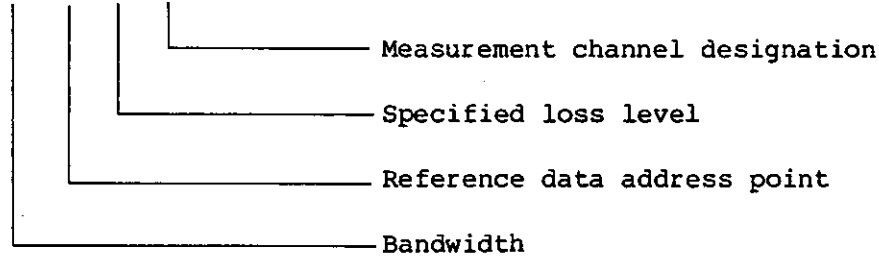
6.3 Description of Built-in Function

6.3.4 Band Width Calculation Function

(1) Function Determining the Minimum Response Value - BND, CBND

BND function : If the reference data address point and LOSS level are specified, the bandwidth is calculated and returned.

BND(P, X, M)

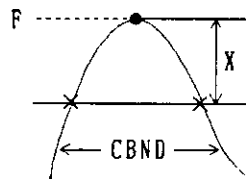
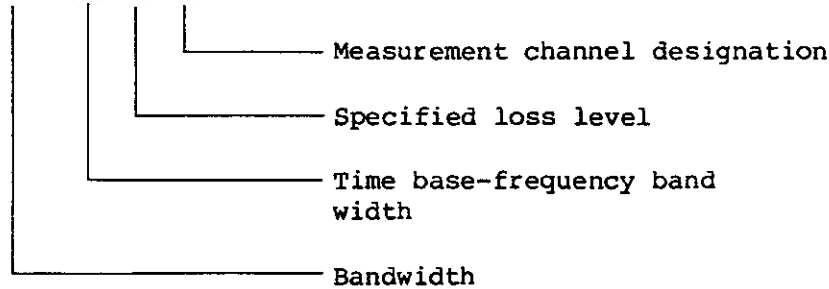


- M=0 (1CH)) Data
- M=1 (2CH))
- M=2 (1CH)) Memory
- M=3 (2CH))

(COMPENSATE)

CBND function : If the time base-frequency and LOSS level are specified, the bandwidth is calculated and returned.

CBND(F, X, M)



- M=0 (1CH)) Data
- M=1 (2CH))
- M=2 (1CH)) Memory
- M=3 (2CH))

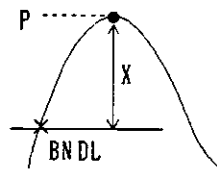
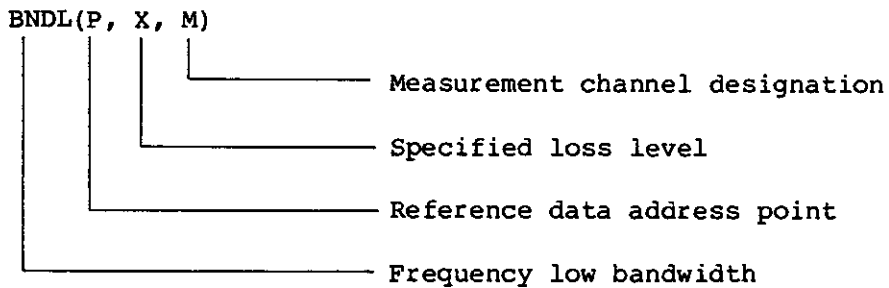
(COMPENSATE)

Note : For polarity of specified loss level, see 6.3.4.

6.3 Description of Built-in Function

(2) Function Determining Frequency with Lower Bandwidth - BNDL, CBNDL

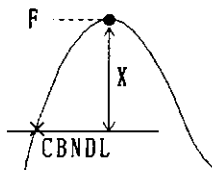
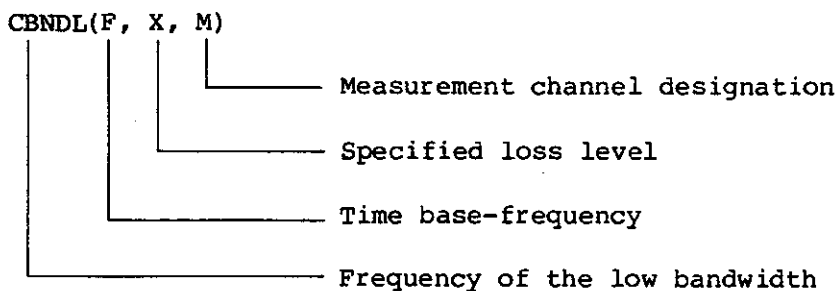
BNDL function : If the reference data address point and LOSS level are specified, the low frequency of the bandwidth is searched for and returned.



M=0 (1CH) Data
M=1 (2CH))
M=2 (1CH) Memory
M=3 (2CH))

(COMPENSATE)

CBNDL function : If time base-frequency and LOSS level are specified, the frequency of the low bandwidth is searched for and returned.



M=0 (1CH) Data
M=1 (2CH))
M=2 (1CH) Memory
M=3 (2CH))

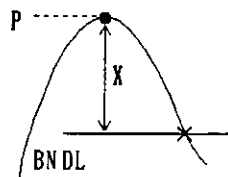
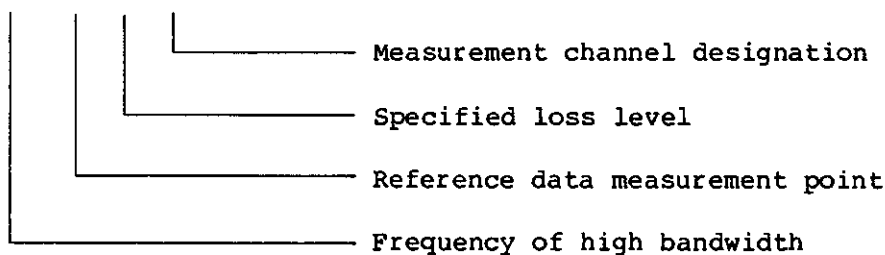
(COMPENSATE)

Note : For polarity of specified loss level, see 6.3.4.

(3) Function Determining Frequency with Upper Bandwidth - BNDH, CBNDH

BNDH function : If the reference data address point and LOSS level are specified, the high frequency of the band width is searched for and returned.

BNDH(P, X, M)

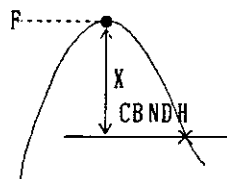
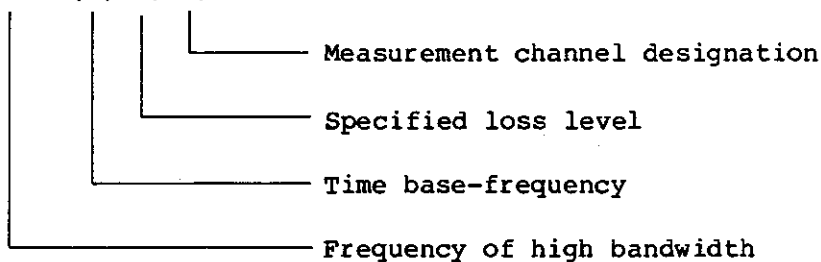


M=0 (1CH)) Data
M=1 (2CH))
M=2 (1CH)) Memory
M=3 (2CH))

(COMPENSATE)

CBNDH function : If time base-frequency and specified LOSS level are specified, the high frequency of the band width is searched for and returned.

CBNDH(F, X, M)



M=0 (1CH)) Data
M=1 (2CH))
M=2 (1CH)) Memory
M=3 (2CH))

(COMPENSATE)

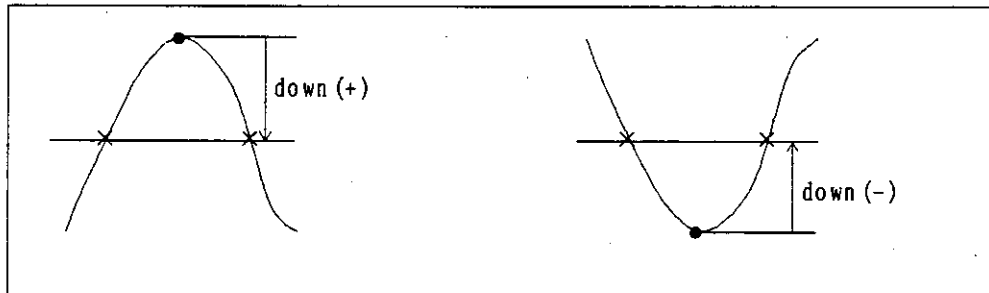
Note : For polarity of specified loss level, see 6.3.4.

NETWORK ANALYZER
PROGRAMMING MANUAL

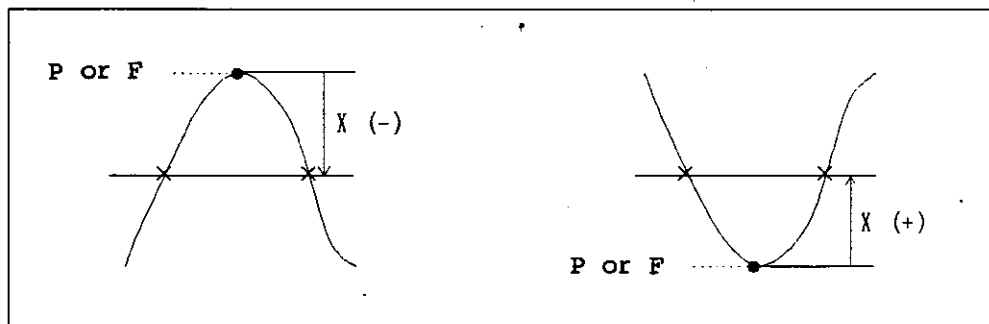
6.3 Description of Built-in Function

Note : The specified LOSS level of BND, CBND, BNDL, CBNDL, BNDH, and CBNDH are handle the following signs.

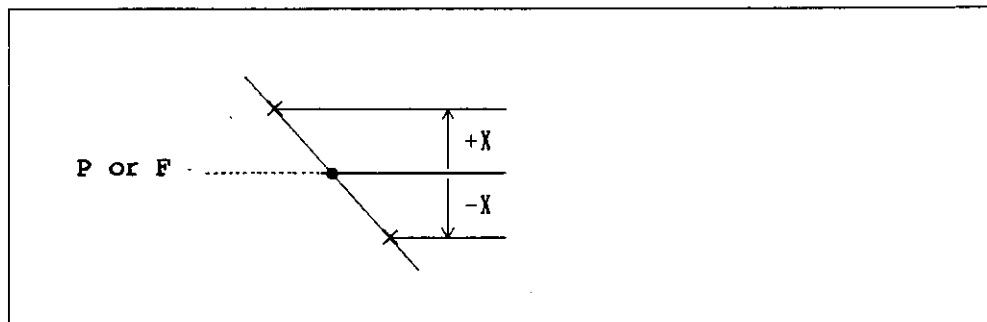
- When FORMAT is LOG MAG;



- When FORMAT is G. DELAY;
(Be careful of polarity. It is the inverse of LOG MAG)



- When FORMAT is PHASE and PHASE $(-\infty, +\infty)$;
(It becomes $\pm X^\circ$ search).



6.3 Description of Built-in Function

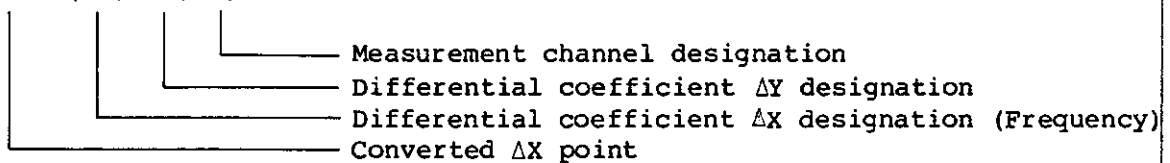
6.3.5 Ripple Function

(1) Differential Coefficient Conversion - DIFFX, DIFFY

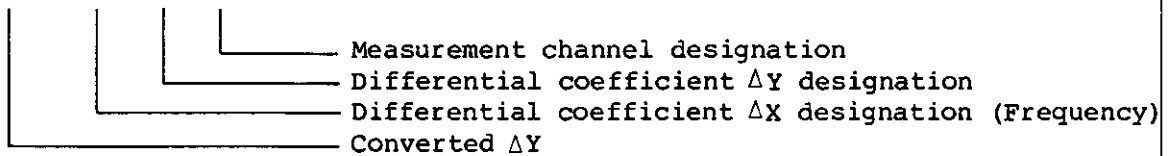
DIFF function : Converts ΔX and ΔY specified by frequency to ΔX and ΔY given by point.

ΔX and ΔY are arguments of built-in function such as ripple, max point, and min point detection.

DIFFX(ΔX , ΔY , M)



DIFFY(ΔX , ΔY , M)

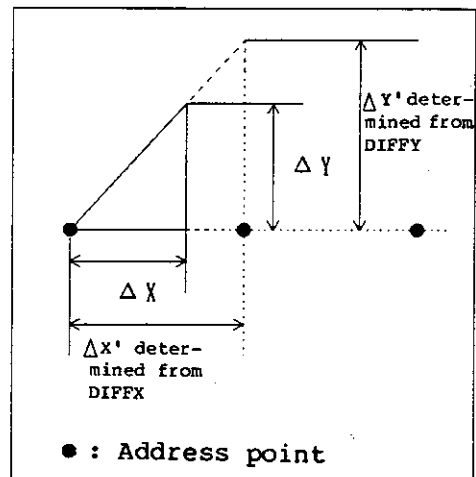


M=0 (1CH)) Data
M=1 (2CH))
M=2 (1CH)) Memory
M=3 (2CH))

Note : Contract for use of same parameters in both functions

(1201 POINT)

Description : DIFFX converts specified ΔX (frequency) to the width of address point. If this point number is used, it is sometimes different from the inclination of ΔX and ΔY . DIFFY is used to determine ΔY for correct inclination (see the figure on the right).



Note : ● When ΔX is negative } Execute after inverting sign
● When ΔY is negative }
● When ΔX is 0 } Error message and -1 are returned
● When ΔY is 0 }

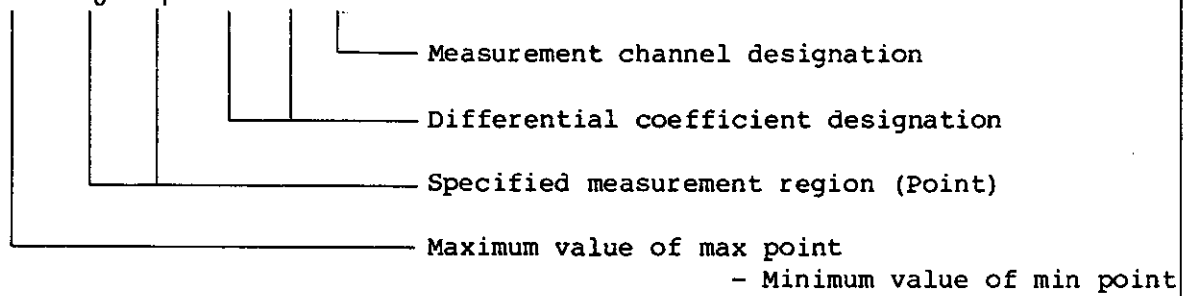
NETWORK ANALYZER
PROGRAMMING MANUAL

6.3 Description of Built-in Function

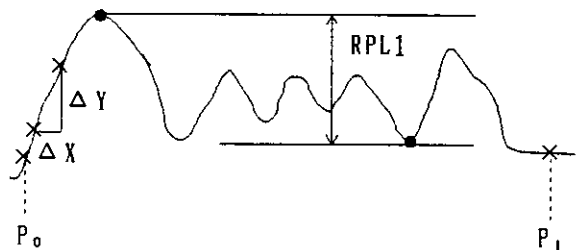
(2) Ripple Detection Function (I) - RPL1, RPL2, RPL3

RPL1 function : If the measurement region address point is specified and if the differential coefficient is specified, a search is made for the maximum and minimum points in that region. The difference between the maximum value and the minimum value is calculated and returned.

RPL1(P₀, P₁, ΔX, ΔY, M)



M=0 (1CH) Data
M=1 (2CH) Data
M=2 (1CH) Memory
M=3 (2CH) Memory



ΔX : Number of points
ΔY : Level (dB)
(Value already converted by
DIFF function)

(1201 POINT)

- Note :
- When ΔX is negative
 - When ΔX is larger than 1200
 - When ΔX is 0
 - When ΔY is negative -----
 - When ΔY is 0 -----

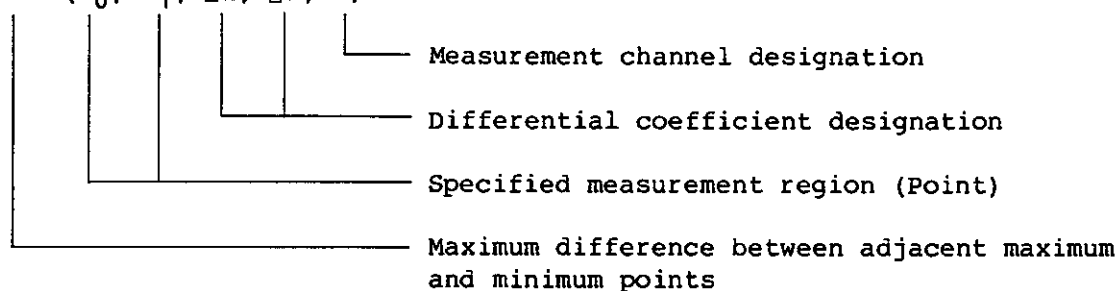
Error message and unspecified value are returned.

Execute after inverting the sign
Error message and unspecified value are returned.

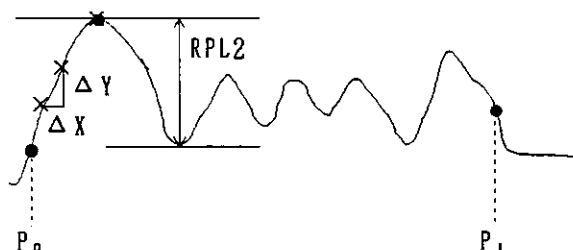
6.3 Description of Built-in Function

RPL2 function : If the measurement region address point is specified and if the differential coefficient is specified, a search is made for the maximum and minimum points in that region. The maximum difference between the adjacent maximum and minimum points is calculated and returned.

RPL2(P₀, P₁, ΔX, ΔY, M)



M=0 (1CH) Data
M=1 (2CH) Data
M=2 (1CH) Memory
M=3 (2CH) Memory



ΔX : Number of points
ΔY : Level (dB)
(Value already converted by
DIFF function)

(1201 POINT)

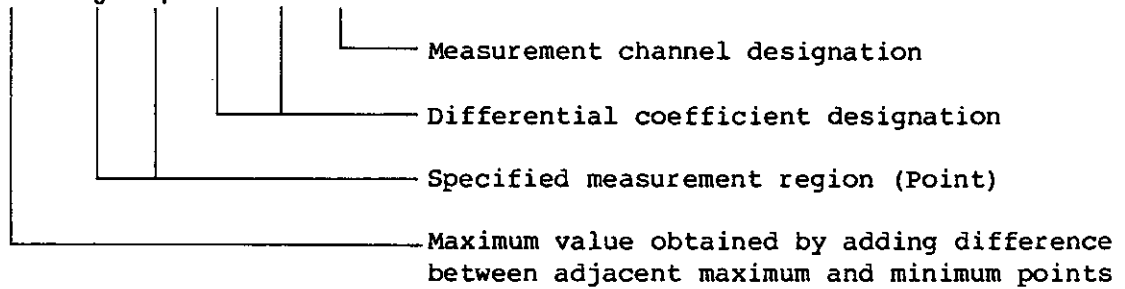
Note : RPL2 is the same note as RPL1.

NETWORK ANALYZER
PROGRAMMING MANUAL

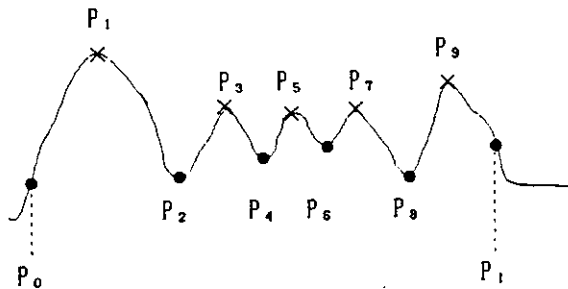
6.3 Description of Built-in Function

RPL3 function : If the measurement region address point is specified and if the differential coefficient is specified, a search is made for the maximum and minimum points in that region. The maximum value obtained by adding the difference between adjacent maximum and minimum points is calculated and returned.

RPL3(P₀, P₁, ΔX, ΔY, M)



M=0 (1CH) Data
M=1 (2CH))
M=2 (1CH) Memory
M=3 (2CH))



ΔX : Number of points
ΔY : Level (dB)
(Value already converted by DIFF function)

(1201 POINT)

$$|(P_2 - P_1) + (P_2 - P_3)|, |(P_4 - P_3) + (P_4 - P_5)|, |(P_6 - P_5) + (P_6 - P_7)|,$$

..... Maximum of these values
(1201 POINT)

Note : RPL3 is the same as RPL1.

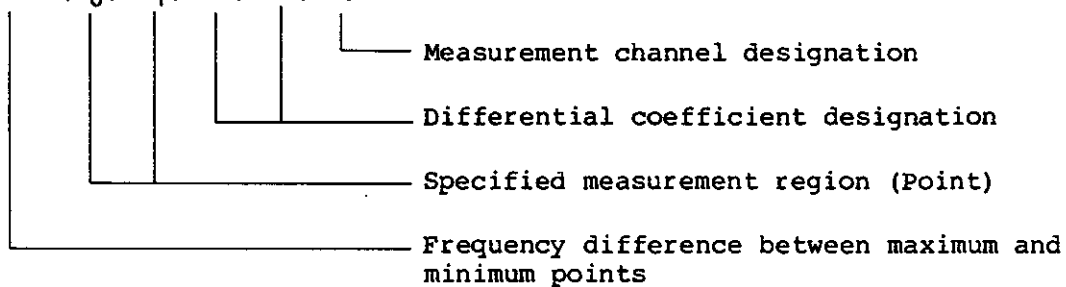
NETWORK ANALYZER
PROGRAMMING MANUAL

6.3 Description of Built-in Function

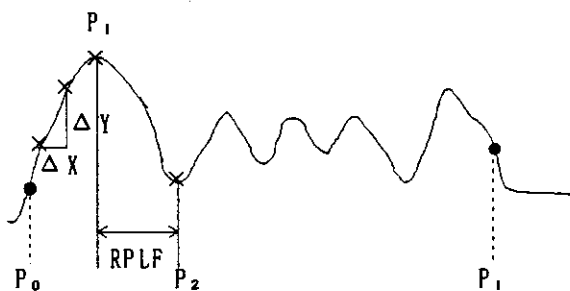
(3) Ripple Detection Function (II) - RPLF, RPLR

RPLF function : If the measurement region address point is specified and if the differential coefficient is specified, a search is made for the maximum and minimum points in that region. The frequency difference between the first maximum and minimum points found is returned.

RPLF(P_0 , P_1 , ΔX , ΔY , M)



M=0 (1CH) Data
M=1 (2CH))
M=2 (1CH) Memory
M=3 (2CH))



ΔX : Number of points
 ΔY : Level (dB)
(Value already converted by
DIFF function)

(1201 POINT)

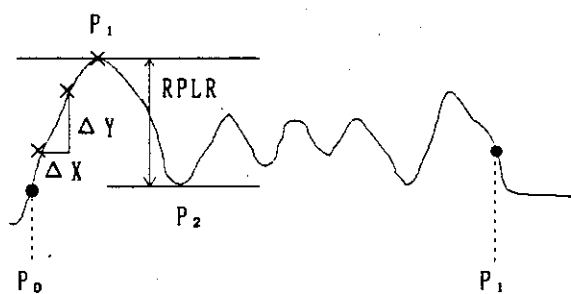
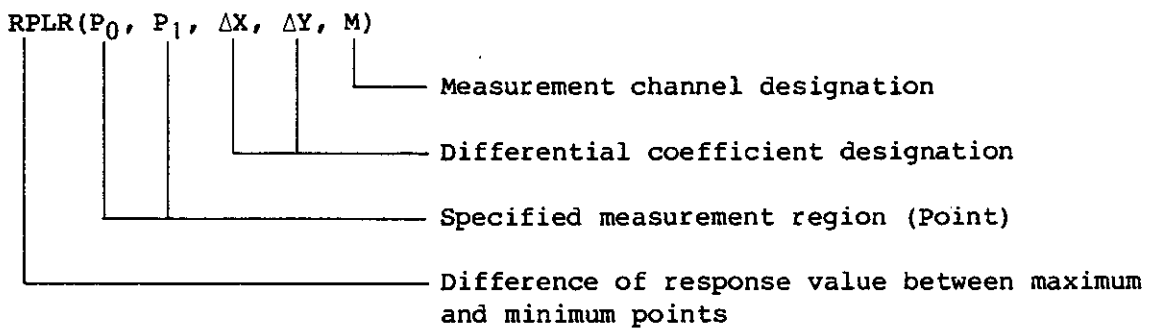
Note : RPLF is the same note as RPL1.

NETWORK ANALYZER
PROGRAMMING MANUAL

6.3 Description of Built-in Function

RPLR function : If the specified region address point is specified and if the differential coefficient is specified, a search is made for the maximum and minimum points in that region. The response value difference between the first maximum and minimum points found is returned.

RPLR(P₀, P₁, ΔX, ΔY, M)



M=0 (1CH)) Data
M=1 (2CH))
M=2 (1CH)) Memory
M=3 (2CH))

Δ X : Number of points
Δ Y : Level (dB)
(Value already converted by
DIFF function)

(1201 POINT)

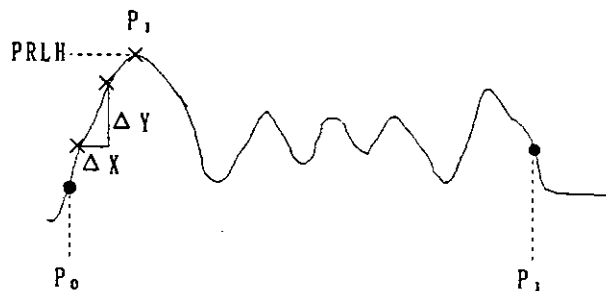
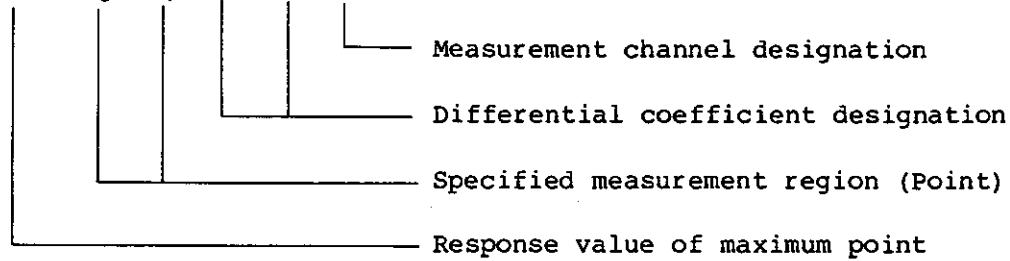
Note : RPLR is the same note as PRL1.

(4) Maximum and Minimum Detection Function

① Function determining maximum and minimum points - RPLH, RPLL

RPLH function : If the specified region address point is specified and if the differential coefficient is specified, a search is made for the maximum and minimum points in that region. The response value of the first maximum point found is returned.

RPLH(P_0 , P_1 , ΔX , ΔY , M)



- M=0 (1CH) Data
- M=1 (2CH) Data
- M=2 (1CH) Memory
- M=3 (2CH) Memory

ΔX : Number of points
 ΔY : Level (dB)
 (Value already converted by DIFF function)

(1201 POINT)

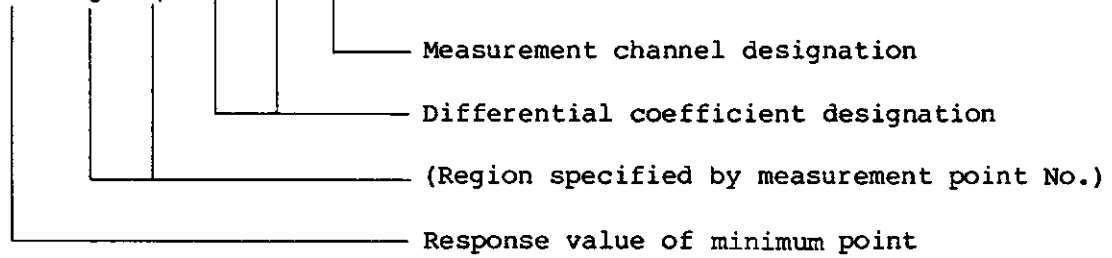
Note : RPLH is the same note as RPL1.

NETWORK ANALYZER
PROGRAMMING MANUAL

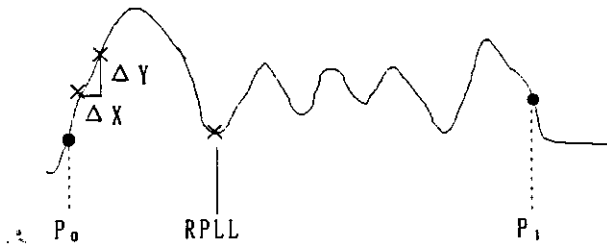
6.3 Description of Built-in Function

RPLL function : If the specified region address point is specified and if the differential coefficient is specified, a search is made for the minimum point in that region. The response value of minimum point found is returned.

RPLL(P₀, P₁, ΔX, ΔY, M)



- M=0 (1CH) Data
- M=1 (2CH) Data
- M=2 (1CH) Memory
- M=3 (2CH) Memory



Δ X : Number of points
Δ Y : Level (dB)
(Value already converted by DIFF function)

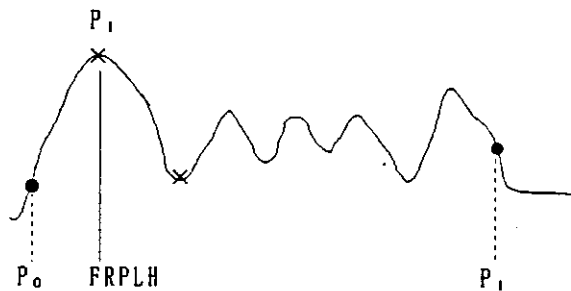
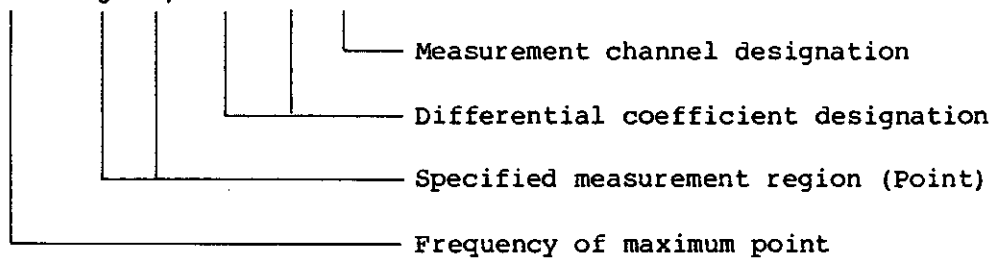
Note : RPLL is the same note as RPL1.

6.3 Description of Built-in Function

- ② Function determining frequency at minimum and maximum points
- FRPLH, FRPLL

FRPLH function : If the specified region address point is specified and if the differential coefficient is specified, a search is made for the maximum point in that region. The frequency of the first maximum point found is returned.

FRPLH(P₀, P₁, ΔX, ΔY, M)



- M=0 (1CH) Data
- M=1 (2CH) Data
- M=2 (1CH) Memory
- M=3 (2CH) Memory

ΔX : Number of points
ΔY : Level (dB)
(Value already converted by
DIFF function)

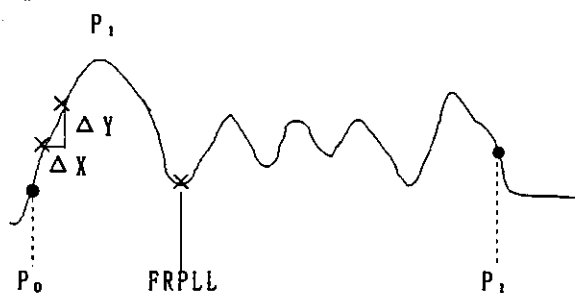
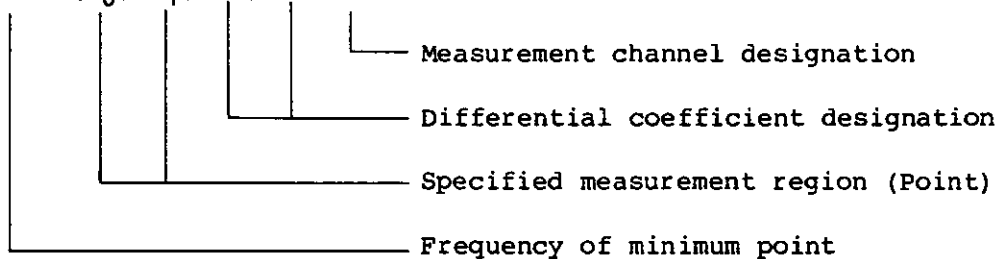
Note : RPLH is the same note as RPL1.

NETWORK ANALYZER
PROGRAMMING MANUAL

6.3 Description of Built-in Function

FRPLL function : If the specified region address point is specified and if the differential coefficient is specified, a search is made for the minimum point in that region. The frequency of the first minimum point found is returned.

FRPLL(P₀, P₁, ΔX, ΔY, M)



M=0 (1CH) Data
M=1 (2CH))
M=2 (1CH) Memory
M=3 (2CH))

ΔX : Number of points
ΔY : Level (dB)
(Value already converted by
DIFF function)

(1201 POINT)

Note : FRPLL is the same note as RPL1.

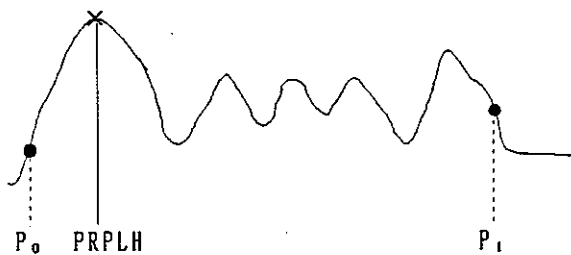
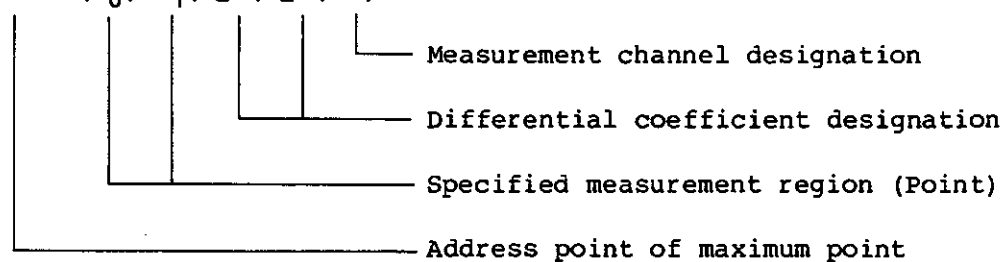
NETWORK ANALYZER
PROGRAMMING MANUAL

6.3 Description of Built-in Function

- ③ Function determining frequency at minimum and maximum points
- FRPLH, FRPLL

PRPLH function : If the specified region address point is specified and if the differential coefficient is specified, a search is made for the maximum points in that region. The address point of the first maximum point found is returned.

PRPLH(P₀, P₁, ΔX, ΔY, M)



Δ X : Number of points
Δ Y : Level (dB)
(Value already converted by
DIFF function)

(1201 POINT)

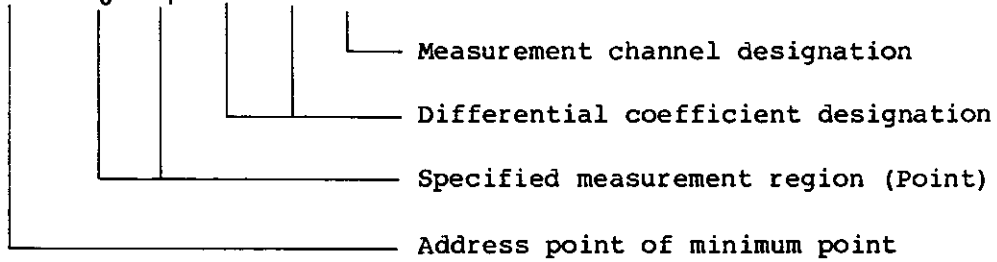
Note : PRPLH is the same note as PRL1. (But error message and -1 are returned if error occurs.)

NETWORK ANALYZER
PROGRAMMING MANUAL

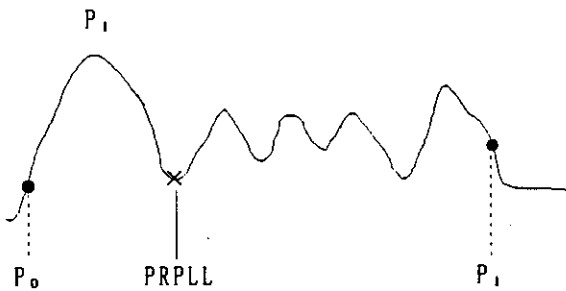
6.3 Description of Built-in Function

PRPLL function : If the specified region address point is specified and if the differential coefficient is specified, a search is made for the minimum point in that region. The address point no. of the first minimum point found is returned.

PRPLL(P₀, P₁, ΔX, ΔY, M)



M=0 (1CH) Data
M=1 (2CH) Data
M=2 (1CH) Memory
M=3 (2CH) Memory



ΔX : Number of points
ΔY : Level (dB)
(Value already converted by
DIFF function)

(1201 POINT)

Note : PRPLL is the same note as RPL1.
(But error message and -1 are returned if error occurs.)

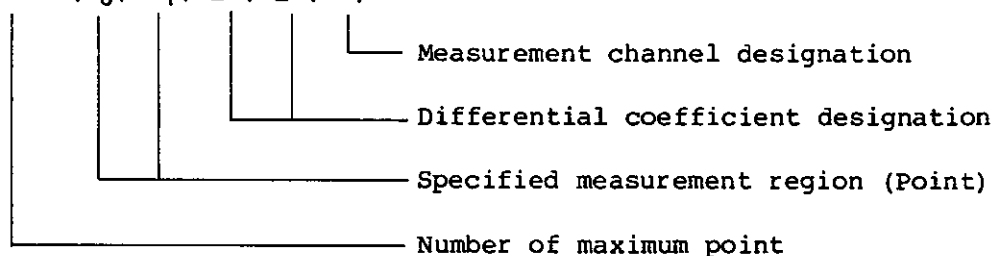
NETWORK ANALYZER
PROGRAMMING MANUAL

6.3 Description of Built-in Function

④ Function determining maximum and minimum points - NRPLH, NRPLL

NRPLH function : If the measurement region address point and the differential coefficient is specified, a search is made for the maximum point in that region. The number of maximum point is determined.

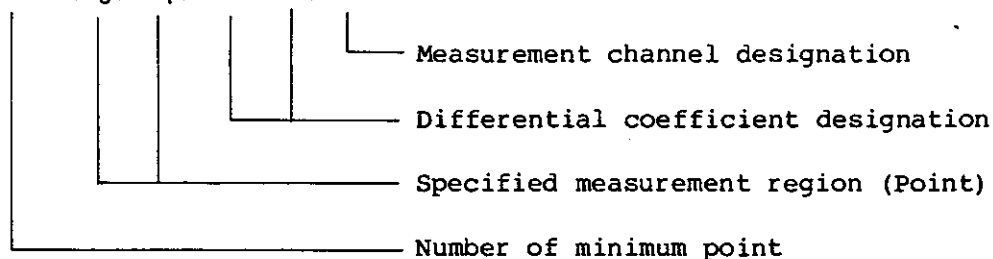
NRPLH(P₀, P₁, ΔX, ΔY, M)



ΔX : Number of points	M=0 (1CH)) Data
ΔY : Level (dB)	M=1 (2CH))
(Value already converted by DIFF function)	M=2 (1CH)) Memory
	M=3 (2CH))

NRPLL function : If the measurement region address point and the differential coefficient are specified, a search is made for the minimum point in that region. The number of minimum point is determined.

NRPLL(P₀, P₁, ΔX, ΔY, M)



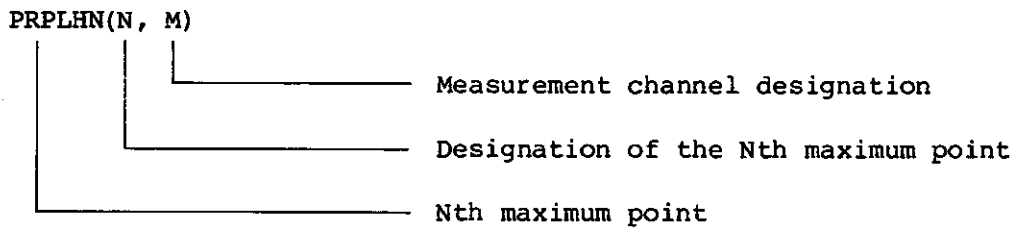
ΔX : Number of points	M=0 (1CH)) Data
ΔY : Level (dB)	M=1 (2CH))
(Value already converted by DIFF function)	M=2 (1CH)) Memory
	M=3 (2CH))

Note : NRPLH and NRPLL are the same note as RPL1.
(But error message and -1 are returned if error occurs.)

6.3 Description of Built-in Function

- ⑤ Function address points at Nth maximum and minimum points
- PRPLHN, PRPLLN

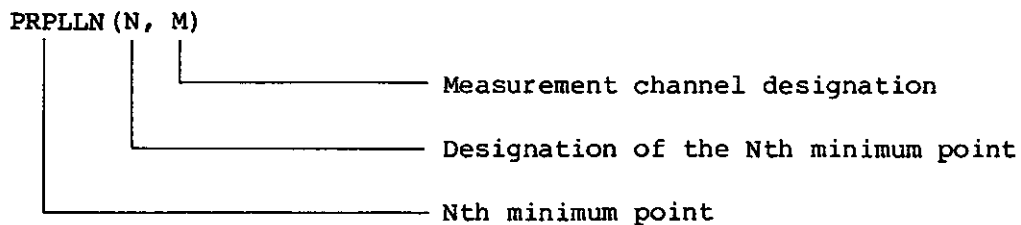
PRPLHN function : Enabled by executing the NRPLH function. Returns the address point for the megalo point if the maximum point number (Nth maximum point) is specified.



Δ X : Number of points
Δ Y : Level (dB)
(Value already converted by DIFF function)

M=0 (1CH)) Data
M=1 (2CH))
M=2 (1CH)) Memory
M=3 (2CH))

PRPLLN function : Enabled by executing the NRPLL function. Returns the address point for the minimum point if the minimum point number (Nth minimum point) is specified.



Δ X : Number of points
Δ Y : Level (dB)
(Value already converted by DIFF function)

M=0 (1CH)) Data
M=1 (2CH))
M=2 (1CH)) Memory
M=3 (2CH))

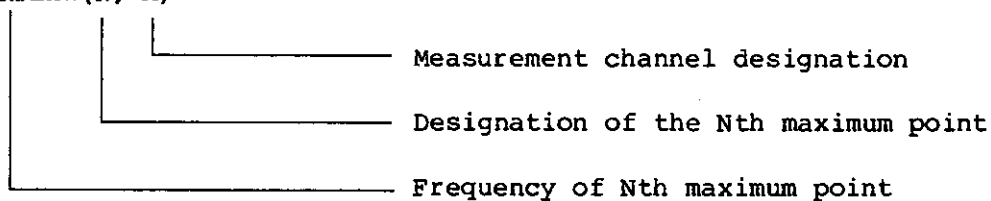
Note : ● When N is not within the range from 1 to [Number of maximum points determined by NRPLH] } Error message and -1 are returned
● When N is not within the range from 1 to [Number of minimum points determined by NRPLL]

6.3 Description of Built-in Function

- ⑥ Function determining frequency at Nth maximum and minimum points
- FRPLHN, FRPLLN

FRPLHN function : If the maximum point no. is specified after executing the NRPLH function, the frequency of that maximum point is displayed.

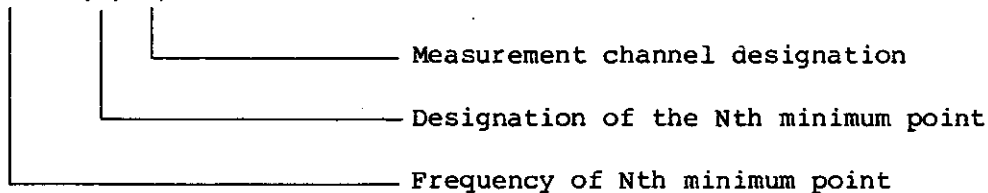
FRPLHN(N, M)



M=0 (1CH)) Data
M=1 (2CH))
M=2 (1CH)) Memory
M=3 (2CH))

FRPLLN function : If the minimum point no. is specified after executing the NRPLL function, the frequency of that minimum point is displayed.

FRPLLN(N, M)



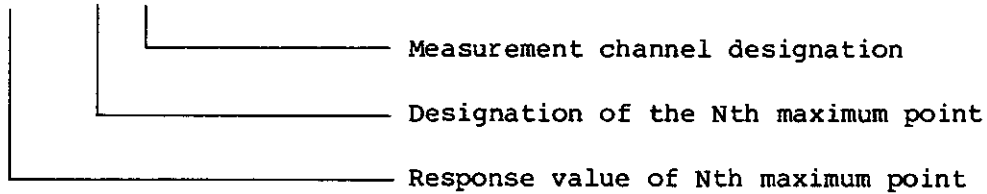
M=0 (1CH)) Data
M=1 (2CH))
M=2 (1CH)) Memory
M=3 (2CH))

Note : ● When N is not within the range from 1 to [Number of maximum values determined by NRPLH] } Error message and -1 are returned
● When N is not within the range from 1 to [Number of minimum values determined by NRPLL] }

- ⑦ Function determining response value at Nth maximum and minimum points
- VRPLHN, VRPLLN

VRPLHN function : Enabled by executing the NRPLF function. Returns the response value for the megalo point if the megalo point number is specified.

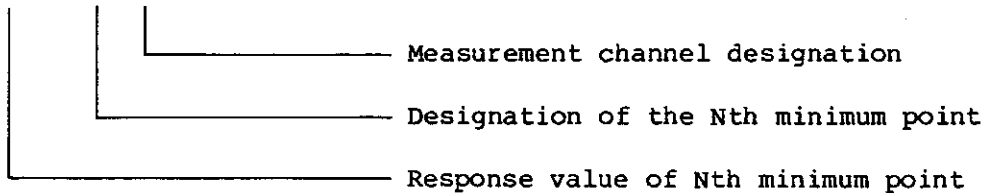
VRPLHN(N, M)



M=0 (1CH)) Data
M=1 (2CH))
M=2 (1CH)) Memory
M=3 (2CH))

VRPLLN function : Enabled by executing the NRPLL function. Returns the response value for the minimum point if the minimum point is specified.

VRPLLN(N, M)



M=0 (1CH)) Data
M=1 (2CH))
M=2 (1CH)) Memory
M=3 (2CH))

Note : Same as for FRPLHN function.

CAUTION

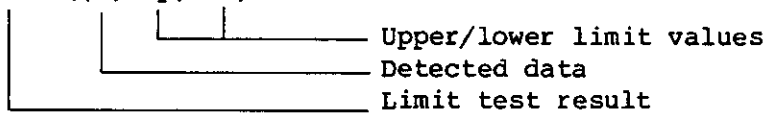
When several ripple functions are used together, no data may be integrated unless (P₀, P₁, ΔX, and ΔY) setting is the same.

6.3.6 Other Functions

(1) Limit Test Function - LMTUL1, LMTUL2, LMTMD1, LMTMD2

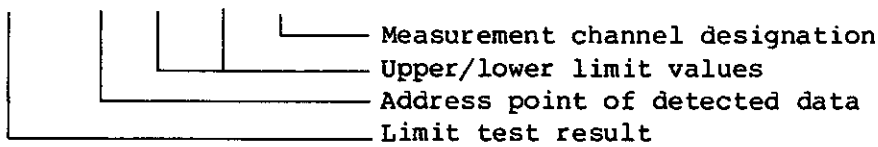
LMT function : If the upper and lower limits, and detected data are given, the fact whether the data lies between the limits or not is checked and the result returned.

LMTUL1(X, Up, Lo)

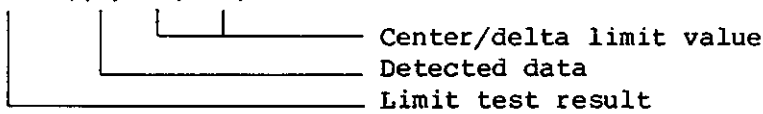


M=0 (1CH) Data
M=1 (2CH) Memory
M=2 (1CH) Data
M=3 (2CH) Memory

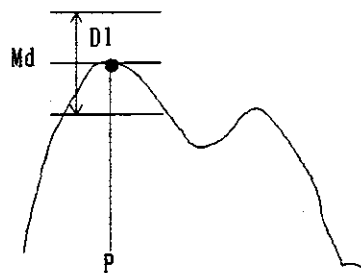
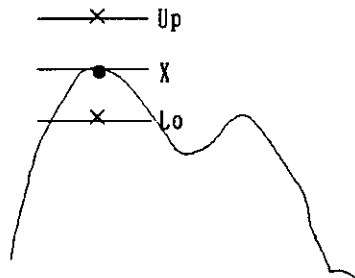
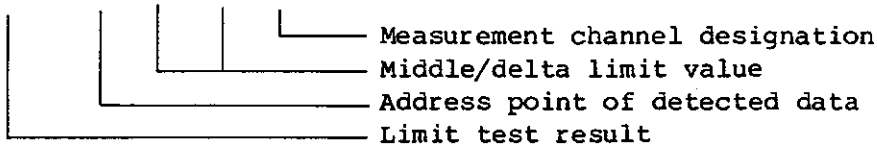
LMTUL2(P, Up, Lo, M)



LMTMD1(X, Md, D1)



LMTMD2(P, Md, D1, M)



(1201 POINT)

Results : When inside range ; 0
When above upper limit ; 1
When below lower limit ; 2
When specified point is not measured after specifying point ;
Return -1

Note : ● When $Lo > Up$ Execute after interchanging Lo and Up
● When $D1$ is negative ... Execute after inverting sign

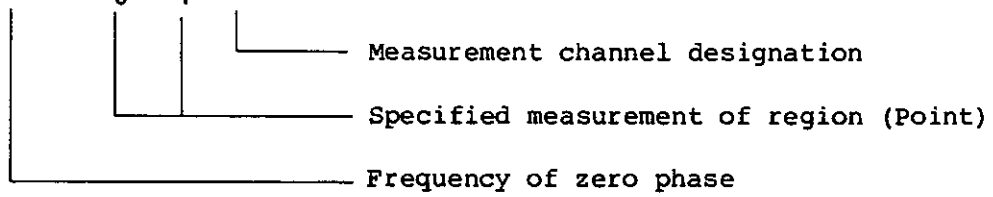
NETWORK ANALYZER
PROGRAMMING MANUAL

6.3 Description of Built-in Function

(2) Zero Phase Detection Function

ZEROPHS function : Zero phase is searched in the specified region by P_0 and P_1 and the frequency is returned.

ZEROPHS(P_0 , P_1 , M)

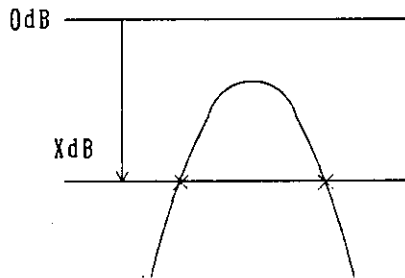
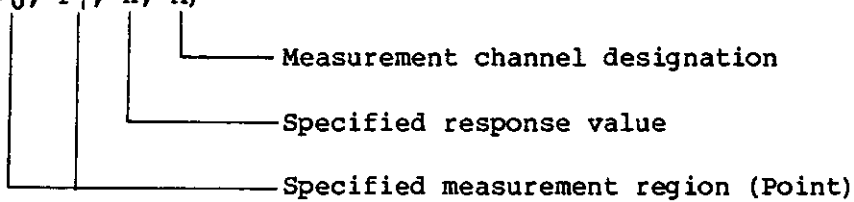


M=0 (1CH)) Data
M=1 (2CH))
M=2 (1CH)) Memory
M=3 (2CH))

(3) Direct Search Functions - DIRECT, CDIRECT, DDIRECT, CDDIRECT

DIRECT function : If the measurement response value is specified, the measurement point of the response value is calculated and returned.

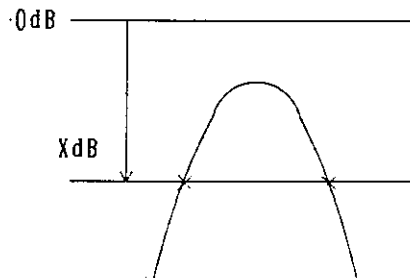
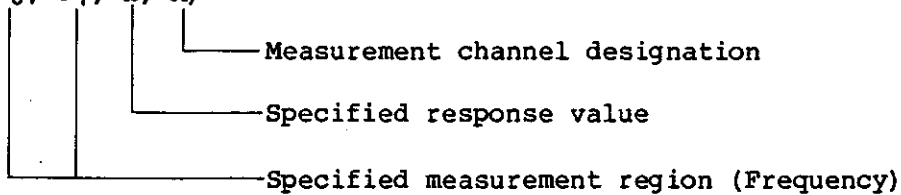
DIRECT(P₀, P₁, X, M)



M=0 (1CH) Data
M=1 (2CH) Data
M=2 (1CH) Memory
M=3 (2CH) Memory

CDIRECT function : If the measurement response value is specified, the frequency of the response value is calculated and returned.

CDIRECT(F₀, F₁, X, M)



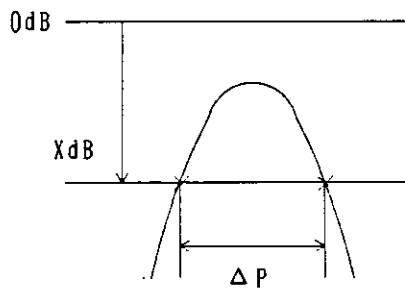
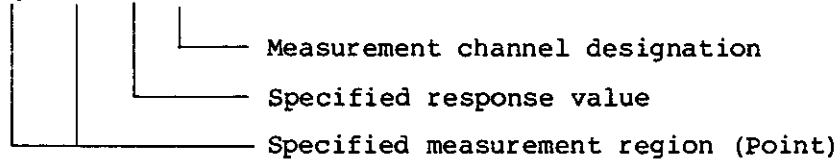
M=0 (1CH) Data
M=1 (2CH) Data
M=2 (1CH) Memory
M=3 (2CH) Memory

Note : ● When P₀=P₁ (F₀=F₁)
● When value X is omitted } An error occurs.

6.3 Description of Built-in Function

DDIRECT function : If the measurement response value is specified, the measurement point difference of the response value is calculated and returned.

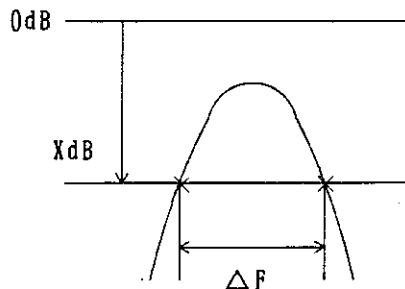
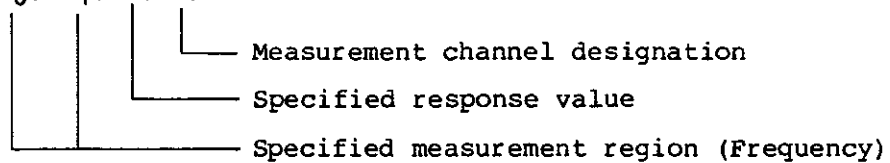
DDIRECT(P_0 , P_1 , X , M)



M=0 (1CH) Data
M=1 (2CH) Data
M=2 (1CH) Memory
M=3 (2CH) Memory

CDDIRECT function : If the measurement response value is specified, the frequency difference of the response value is calculated and returned.

CDDIRECT(F_0 , F_1 , X , M)



M=0 (1CH) Data
M=1 (2CH) Data
M=2 (1CH) Memory
M=3 (2CH) Memory

Note : ● When $P_0=P_1$ ($F_0=F_1$)
● When value X is omitted } An error occurs.

MEMO



A large, empty rectangular area with rounded corners, enclosed by a dashed border, intended for writing the memo's content.

APPENDIX

A.1 Error Message

A.1.1 How to Display an Error Message

- (1) Change the measurement screen to the edit screen.
- (2) Enter [?ERRM\$(0)] in the direct mode.
? is an abbreviation of PRINT.
- (3) An error message and line number causing an error are displayed.

A.1.2 How to Display the Present Position of Program

@ is called system variable, and the line number executing the program is stored. If a system variable is used, the present line number, present program position, and program stop position can be understood.

Example : ? @ → Indicates program stop position.

A.1.3 Error Messages

Note 1 : The following table lists error messages in the alphabetical order.

A character string is represented by XXX.
A numeric value is represented by YYY.

Note 2 : Details of error classification

- 1 : Related to data input/output
- 2 : Related to data operation processing
- 3 : Related to built-in function
- 4 : Related to BASIC statement
- 5 : Others

NETWORK ANALYZER
PROGRAMMING MANUAL

A.1 Error Message

Error class (error No.)	Error message	Description
4(33)	Array's range error	A subscript of array variable exceeds the statement range.
1(28)	Bad free call	Memory control error
4(79)	CANNOT assigned into this token	A character string cannot be substituted into the variable.
1(66)	cannot read data from "xxx" file.	The number of characters specified by xxx file cannot be read.
4(68)	cannot specify "USING"	USING cannot be specified by type of file specified.
1(67)	cannot write data into "xxx" file.	Data cannot be in the xxx file.
1(74)	end of "xxx" file	EOF (End Of File) is read.
4(40)	expression format error	The expression format error occurs.
1(85)	file format error	No terminator is within 256 characters.
1(72)	file is NOT open.	No file is registered in specified descriptor (no file is opened).
4(15)	FOR <init value > does NOT exist.	No initial value of FOR statement exists.
4(14)	FOR variable does NOT exist.	No counter variable of FOR statement exists.
4(13)	FOR's nest is abnormal.	No FOR can be nested.
4(29)	Invalid dimension parameter	A parameter of array variable is incorrect.
2(59)	Invalid string constant	Double quotation cannot be set.
4(38)	label not found	No specified label
4(19)	Label xxx is already exists.	The xxx label exists already.
2(56)	Line No. yyy is out of range.	Specified line number is out of program range.
1(30)	memory space full	No memory space

NETWORK ANALYZER
PROGRAMMING MANUAL

A.1 Error Message

Error class (error No.)	Error message	Description
4(3)	NO operand in xxx	Operand of xxx is incorrect.
4(21)	Not available ASCII char (yyy)	No ASCII code is available.
4(70)	Not found DATA statement	No DATA statement is found in RESTORE.
4(45)	Not found THEN in xxx	No THEN is at the end of IF statement.
1(81)	Only one INPUT file can be opened.	More than one INPUT file is to be opened in the read mode.
1(76)	Only one OUTPUT file can be opened.	More than one OUTPUT file is to be opened in the write mode.
2(51)	Overflow value	An overflow occurs.
4(50)	parameter error	A parameter error occurs.
5(55)	Program CANNOT be continued.	A program cannot be continued.
4(5)	Program is NOT exist	No program exists.
7(78)	SELECT nesting overflow	Too many nests in the SELECT statement
4(31)	string declaration error	"[" "]" is used for numeric variable.
2(32)	string length is too long	Declaration of character string variable is too long. (Max. 128)
2(49)	Substring error	A substring error occurs.
4(17)	Unbalanced BREAK	No BREAK statement is between FOR and NEXT.
4(16)	Unbalanced FOR variable in NEXT	Relation between FOR and NEXT statement is abnormal.
4(34)	Unbalanced line No.	No specified line
4(12)	Unbalanced NEXT statement	No NEXT statement is found even though the FOR statement is done.
4(20)	Unbalanced xxx	Statement is unbalanced.

NETWORK ANALYZER
PROGRAMMING MANUAL

A.1 Error Message

Error class (error No.)	Error message	Description
4(44)	Unbalanced xxx block	The xxx block (such as FOR and IF statements) is unbalanced.
4(37)	Undefined label	No specified label
4(7)	undefined ON condition	No ON condition is undefined, but the ON state is found.
4(18)	Uninstalled type (xxx)	A variable format is abnormal.
4(39)	Unknown line No.	No specified line
2(63)	Unmatched DATA's values and READ variable	Data to be read with the READ statement is not found.
4(52)	Unmatched IMAGE-spec in USING	Specification of USING image is unmatched.
4(86)	You cannot use xxx command	No xxx command can be used.
3(11)	xxx function error	A built-in function error occurs.
4(71)	xxx nest overflow	A nest overflow occurs.
1(22)	xxx1 (xxx2) error	The xxx1 instruction cannot be executed to the xxx2 file.
1(23)	xxx1 (xxx2, xxx3) error	The xxx1 instruction cannot be executed to the xxx2 and xxx3 files.
1(65)	xxx : "xxx" file was opened with xxx mode	Access mode is different from that of opening file.
2(10)	xxx : CANNOT convert into string	Cannot be converted into character string.
4(24)	xxx : invalid first type in xxx	The first statement of command is invalid.
4(25)	xxx : invalid second type in xxx	The second statement of command is invalid.
4(26)	xxx : invalid source type in xxx	Source type is invalid when the expression is substituted.

NETWORK ANALYZER
PROGRAMMING MANUAL

A.1 Error Message

Error class (error No.)	Error message	Description
4(27)	xxx : invalid target type in xxx	Type of variable to be substituted is invalid.
4(9)	xxx : Invalid TARGET operand in xxx	An invalid TARGET is found in xxx.
4(2)	xxx : invalid type in xxx	An invalid type is found in xxx.
4(6)	xxx : Syntax error	A syntax error occurs.
1(64)	"xxx" file cannot be opened.	No file to be opened
1(69)	"xxx" file is already opened with another PATH.	The file already opened is to be opened.
1(75)	"xxx" file is already exist.	The existed file is to be opened in the OUTPUT mode.
4(82)	"xxx" read error.	Read error
4(54)	yyy error(s) appeared.	An error occurs in the label No.
2(43)	yyy is invalid value in xxx	xxx instruction makes yyy invalid.
2(41)	yyy : UNIT addr error in xxx	An error occurs in specification of GPIB address.
2(60)	yyy : Undefined Control Register	Registration of CONTROL instruction is abnormal.
2(2)	0 divide	Division is done by zero (n/0).

MEMO



A large, empty rectangular area with rounded corners, enclosed by a thin black border, intended for writing the memo's content.

NETWORK ANALYZER
PROGRAMMING MANUAL

ALPHABETICAL INDEX

ALPHABETICAL INDEX (Cont'd)

FRPLHN function 6 - 34
 FRPLL function 6 - 29
 FRPLLN function 6 - 34
 File Deletion 3 - 9
 File Management 3 - 7
 File Management 3 - 8
 File Name Change 3 - 9
 File Recalling 3 - 9
 File Storage 3 - 9
 File control statements 5 - 4
 File type 3 - 7
 Floppy Disk 3 - 3
 Floppy Disk Component Parts . 3 - 3
 Floppy Disk Dimensions 3 - 3
 Floppy Disk Insertion Method 3 - 4
 Full name 4 - 15
 Function Operations 4 - 8
 Functions 4 - 20

[G]

GLIST 5 - 16
 GLISTN 5 - 18
 GOSUB 5 - 51
 GOTO 5 - 53
 GPIB Addressing 2 - 3
 GPIB Code Table 2 - 9
 GPIB EXTERNAL CONTROLLER 2 - 1
 GPIB Functions 2 - 2
 GPIB Input and Output Formats 2 - 5
 GPIB Modes 1 - 2
 GPIB Program Code 2 - 11
 GPIB control statements 5 - 4
 GPRINT 5 - 54
 Generation of Program List .. 4 - 12

[H]

Head window 3 - 3

[I]

IF THEN 5 - 55
 INIT 5 - 20
 INITIALIZE 5 - 20
 INPUT 5 - 58
 INTEGER 5 - 60
 INTERFACE CLEAR 5 - 90

Initialization of Floppy Disk 3 - 8
 Input 4 - 3
 Input Formats 2 - 8
 Input of Program Lines 4 - 11
 Input type 2 - 5
 Insertion of Characters 4 - 11
 Insertion of Lines 4 - 11

[K]

Key Words 4 - 15

[L]

LET 5 - 62
 LIMIT LINE (OUTPUT) 2 - 36
 LIST 5 - 21
 LISTN 5 - 23
 LLIST 5 - 25
 LLISTN 5 - 27
 LMT function 6 - 36
 LOAD 5 - 29
 LOCAL 5 - 91
 LOCAL LOCKOUT 5 - 92
 LPRINT 5 - 54
 Label 3 - 3
 Limited Test Function Is
 Used in Low-pass Filter
 Measurements 2 - 54
 Logical Operators 4 - 25

[M]

MAX function 6 - 13
 MERGE 5 - 30
 MIN function 6 - 14
 Measuring Point 6 - 1
 Measuring Program Using
 Parallel I/O Ports 2 - 50

[N]

NEXT 5 - 49
 NRPLH function 6 - 32
 NRPLL function 6 - 32

[O]

OFF END 5 - 115
 OFF ISRQ 5 - 65

NETWORK ANALYZER
PROGRAMMING MANUAL

ALPHABETICAL INDEX

ALPHABETICAL INDEX (Cont'd)

OFF KEY	5 - 64	REM	5 - 81
OFF SRQ	5 - 65	REMOTE	5 - 95
ON END	5 - 116	REMOTE CONTROL	2 - 1
ON ERROR	5 - 66	REN	5 - 32
ON ISRQ	5 - 69	RENAME	5 - 33
ON KEY	5 - 67	REQUEST	5 - 96
ON SRQ	5 - 69	RESTORE	5 - 82
OPEN	5 - 117	RETURN	5 - 51
OUT	5 - 119	RPL1 function	6 - 21
OUTPUT	5 - 93	RPL2 function	6 - 22
OUTPUT	5 - 119	RPL3 function	6 - 23
OUTPUT (OUT) USING	5 - 121	RPLF function	6 - 24
Objects	4 - 16	RPLH function	6 - 26
Operators	4 - 23	RPLL function	6 - 27
Output Format	2 - 7	RPLR function	6 - 25
		RUN	5 - 34
		Rearranging Program Numbers .	4 - 12
[P]		Recalling Programs	3 - 7
PAUSE	5 - 71	Response to Built-in Function	6 - 1
PEEK	5 - 72	Response type	2 - 6
PMAX function	6 - 14		
PMIN function	6 - 15	[S]	
POINT function	6 - 8	SAVE	5 - 35
POKE	5 - 73	SCRATCH	5 - 36
PRINT	5 - 74	SECTOR	3 - 7
PRINT USING	5 - 74	SELECT	5 - 84
PRINTER	5 - 31	SEND	5 - 97
PRINTER	5 - 77	SPOLL	5 - 99
PRINTF	5 - 78	SPRINTF	5 - 83
PRPLH function	6 - 30	SRQ	2 - 37
PRPLHN function	6 - 33	STEP	5 - 37
PRPLL function	6 - 31	Saving Programs	3 - 7
PRPLLN function	6 - 33	Service Request	2 - 31
PURGE	5 - 31	Setting Controller Mode	3 - 2
Permissible Input Characters	2 - 6	Shorten name	4 - 15
Program Architecture	4 - 14	Starting BASIC from	
Program Editing	4 - 11	External Controller	2 - 39
Program Editor Keys	4 - 6	Statements	4 - 14
Program Example	2 - 45	Statements	5 - 2
Program Examples	2 - 32	Status Register	2 - 31
Program Mode	4 - 2	Sub String Operator	4 - 26
Programming Rules	4 - 14	Substitution Operators	4 - 23
Programs	4 - 3	Syntax of BASIC File	
		Control Statement	5 - 101
[R]		System Controller Mode	1 - 2
READ	5 - 80		

NETWORK ANALYZER
PROGRAMMING MANUAL

ALPHABETICAL INDEX

ALPHABETICAL INDEX (Cont'd)

[T]

TALKER/LISTENER Mode 1 - 2
TRACE DATA (INPUT) 2 - 34
TRACE DATA (OUTPUT) 2 - 35
TRIGGER 5 - 100

[U]

Unary Arithmetic Operators .. 4 - 24

[V]

VALUE function 6 - 11
VRPLHN function 6 - 35
VRPLLN function 6 - 35
Variables 4 - 18

[W]

Write Protect 3 - 6
Write protection hole 3 - 3

[X]

X'TAL Filter Measuring
Program 2 - 47

[Z]

ZEROPHS function 6 - 37

IMPORTANT INFORMATION FOR ADVANTEST SOFTWARE

PLEASE READ CAREFULLY: This is an important notice for the software defined herein. Computer programs including any additions, modifications and updates thereof, operation manuals, and related materials provided by Advantest (hereafter referred to as "SOFTWARE"), included in or used with hardware produced by Advantest (hereafter referred to as "PRODUCTS").

SOFTWARE License

All rights in and to the SOFTWARE (including, but not limited to, copyright) shall be and remain vested in Advantest. Advantest hereby grants you a license to use the SOFTWARE only on or with Advantest PRODUCTS.

Restrictions

- (1) You may not use the SOFTWARE for any purpose other than for the use of the PRODUCTS.
- (2) You may not copy, modify, or change, all or any part of, the SOFTWARE without permission from Advantest.
- (3) You may not reverse engineer, de-compile, or disassemble, all or any part of, the SOFTWARE.

Liability

Advantest shall have no liability (1) for any PRODUCT failures, which may arise out of any misuse (misuse is deemed to be use of the SOFTWARE for purposes other than its intended use) of the SOFTWARE. (2) For any dispute between you and any third party for any reason whatsoever including, but not limited to, infringement of intellectual property rights.

LIMITED WARRANTY

1. Unless otherwise specifically agreed by Seller and Purchaser in writing, Advantest will warrant to the Purchaser that during the Warranty Period this Product (other than consumables included in the Product) will be free from defects in material and workmanship and shall conform to the specifications set forth in this Operation Manual.
2. The warranty period for the Product (the "Warranty Period") will be a period of one year commencing on the delivery date of the Product.
3. If the Product is found to be defective during the Warranty Period, Advantest will, at its option and in its sole and absolute discretion, either (a) repair the defective Product or part or component thereof or (b) replace the defective Product or part or component thereof, in either case at Advantest's sole cost and expense.
4. This limited warranty will not apply to defects or damage to the Product or any part or component thereof resulting from any of the following:
 - (a) any modifications, maintenance or repairs other than modifications, maintenance or repairs (i) performed by Advantest or (ii) specifically recommended or authorized by Advantest and performed in accordance with Advantest's instructions;
 - (b) any improper or inadequate handling, carriage or storage of the Product by the Purchaser or any third party (other than Advantest or its agents);
 - (c) use of the Product under operating conditions or environments different than those specified in the Operation Manual or recommended by Advantest, including, without limitation, (i) instances where the Product has been subjected to physical stress or electrical voltage exceeding the permissible range and (ii) instances where the corrosion of electrical circuits or other deterioration was accelerated by exposure to corrosive gases or dusty environments;
 - (d) use of the Product in connection with software, interfaces, products or parts other than software, interfaces, products or parts supplied or recommended by Advantest;
 - (e) incorporation in the Product of any parts or components (i) provided by Purchaser or (ii) provided by a third party at the request or direction of Purchaser or due to specifications or designs supplied by Purchaser (including, without limitation, any degradation in performance of such parts or components);
 - (f) Advantest's incorporation or use of any specifications or designs supplied by Purchaser;
 - (g) the occurrence of an event of force majeure, including, without limitation, fire, explosion, geological change, storm, flood, earthquake, tidal wave, lightning or act of war; or
 - (h) any negligent act or omission of the Purchaser or any third party other than Advantest.
5. **EXCEPT TO THE EXTENT EXPRESSLY PROVIDED HEREIN, ADVANTEST HEREBY EXPRESSLY DISCLAIMS, AND THE PURCHASER HEREBY WAIVES, ALL WARRANTIES, WHETHER EXPRESS OR IMPLIED, STATUTORY OR OTHERWISE, INCLUDING, WITHOUT LIMITATION, (A) ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE AND (B) ANY WARRANTY OR REPRESENTATION AS TO THE VALIDITY, SCOPE, EFFECTIVENESS OR USEFULNESS OF ANY TECHNOLOGY OR ANY INVENTION.**
6. **THE REMEDY SET FORTH HEREIN SHALL BE THE SOLE AND EXCLUSIVE REMEDY OF THE PURCHASER FOR BREACH OF WARRANTY WITH RESPECT TO THE PRODUCT.**
7. **ADVANTEST WILL NOT HAVE ANY LIABILITY TO THE PURCHASER FOR ANY INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL OR PUNITIVE DAMAGES, INCLUDING, WITHOUT LIMITATION, LOSS OF ANTICIPATED PROFITS OR REVENUES, IN ANY AND ALL CIRCUMSTANCES, EVEN IF ADVANTEST HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES AND WHETHER ARISING OUT OF BREACH OF CONTRACT, WARRANTY, TORT (INCLUDING, WITHOUT LIMITATION, NEGLIGENCE), STRICT LIABILITY, INDEMNITY, CONTRIBUTION OR OTHERWISE. TORT (INCLUDING, WITHOUT LIMITATION, NEGLIGENCE), STRICT LIABILITY, INDEMNITY, CONTRIBUTION OR OTHERWISE.**
8. **OTHER THAN THE REMEDY FOR THE BREACH OF WARRANTY SET FORTH HEREIN, ADVANTEST SHALL NOT BE LIABLE FOR, AND HEREBY DISCLAIMS TO THE FULLEST EXTENT PERMITTED BY LAW ANY LIABILITY FOR, DAMAGES FOR PRODUCT FAILURE OR DEFECT, WHETHER ARISING OUT OF BREACH OF CONTRACT, TORT (INCLUDING, WITHOUT LIMITATION, NEGLIGENCE), STRICT LIABILITY, INDEMNITY, CONTRIBUTION OR OTHERWISE.**

CUSTOMER SERVICE DESCRIPTION

In order to maintain safe and trouble-free operation of the Product and to prevent the incurrence of unnecessary costs and expenses, Advantest recommends a regular preventive maintenance program under its maintenance agreement.

Advantest's maintenance agreement provides the Purchaser on-site and off-site maintenance, parts, maintenance machinery, regular inspections, and telephone support and will last a maximum of ten years from the date the delivery of the Product. For specific details of the services provided under the maintenance agreement, please contact the nearest Advantest office listed at the end of this Operation Manual or Advantest's sales representatives.

Some of the components and parts of this Product have a limited operating life (such as, electrical and mechanical parts, fan motors, unit power supply, etc.). Accordingly, these components and parts will have to be replaced on a periodic basis. If the operating life of a component or part has expired and such component or part has not been replaced, there is a possibility that the Product will not perform properly. Additionally, if the operating life of a component or part has expired and continued use of such component or part damages the Product, the Product may not be repairable. Please contact the nearest Advantest office listed at the end of this Operation Manual or Advantest's sales representatives to determine the operating life of a specific component or part, as the operating life may vary depending on various factors such as operating condition and usage environment.

SALES & SUPPORT OFFICES

Advantest Korea Co., Ltd.

22BF, Kyobo KangNam Tower,
1303-22, Seocho-Dong, Seocho-Ku, Seoul #137-070, Korea
Phone: +82-2-532-7071
Fax: +82-2-532-7132

Advantest (Suzhou) Co., Ltd.

Shanghai Branch Office:
Bldg. 6D, NO.1188 Gumei Road, Shanghai, China 201102 P.R.C.
Phone: +86-21-6485-2725
Fax: +86-21-6485-2726

Shanghai Branch Office:
406/F, Ying Building, Quantum Plaza, No. 23 Zhi Chun Road,
Hai Dian District, Beijing,
China 100083
Phone: +86-10-8235-3377
Fax: +86-10-8235-6717

Advantest (Singapore) Pte. Ltd.

438A Alexandra Road, #08-03/06
Alexandra Technopark Singapore 119967
Phone: +65-6274-3100
Fax: +65-6274-4055

Advantest America, Inc.

3201 Scott Boulevard, Suite, Santa Clara, CA 95054, U.S.A
Phone: +1-408-988-7700
Fax: +1-408-987-0691

ROHDE & SCHWARZ Europe GmbH

Mühldorfstraße 15 D-81671 München, Germany
(P.O.B. 80 14 60 D-81614 München, Germany)
Phone: +49-89-4129-13711
Fax: +49-89-4129-13723

ADVANTEST®

<http://www.advantest.co.jp>