HP 3245A Universal Source

HP 3245A Command Reference Manual

03245-90002

HEWLETT PACKARD

Manual Part Number: 03245-90002
Microfiche Part Number: 03245-99002

Printed: SEPTEMBER 1988 Edition 1
Printed in U.S.A. E0988

Printing History

The Printing History shown below lists the printing dates of all Editions and Updates created for this manual. The Edition number changes as the manual undergoes subsequent revisions. Editions are numbered sequentially starting with Edition 1.

Updates, which are issued between Editions, contain individual replacement pages which the customer uses to update the current Edition of the manual. Updates are numbered sequentially starting with Update 1. When a new Edition is created, all Updates associated with the previous Edition are merged into the manual. Each new Edition or Update also includes a revised copy of this printing history page.

Many product updates and revisions do not require manual changes and, conversely, manual corrections may be done without accompanying product changes. Therefore, do not expect a one-to-one correspondence between product updates and manual updates.

Edition 1 (Part Number 03245-90002) .... SEPTEMBER 1988

HEWLETT PACKARD

Herstellerbescheinigung

Hiermit wird bescheinigt, daβ das Gerät/System HP 3245A in Übereinstimmung mit den Bestimmungen von Postverfügung 1046/84 funkentstört ist.

Der Deutschen Bundespost wurde das Inverkehrbringen dieses Gerätes/Systems angezeigt und die Berechtigung zur Überprüfung der Serie auf Einhaltung der Bestimmungen eingeräumt.

Zusatzinformation fur Meβ- und Testgeräte

Werden Meβ- und Testgeräte mit ungeschirmten Kabeln und/oder in offenen Meβaufbauten verwendet, so ist vom Betreiber sicherzustellen, daβ die Funk-Entstörbestimmungen unter Betriebsbedingungen an seiner Grundstücksgrenze eingehalten werden.

Manufacturer's declaration

This is to certify that the equipment HP 3245A is in accordance with the Radio Interference Requirements of Directive FTZ 1046/84. The German Bundespost was notified that this equipment was put into circulation, the right to check the series for compliance with the requirements was granted.

Additional Information for Test- and Measurement Equipment

If Test- and Measurement Equipment is operated with unscreened cables and/or used for measurements on open set-ups, the user has to assure that under operating conditions the Radio Interference Limits are still met at the border of his premises.

HEWLETT PACKARD

CERTIFICATION

Hewlett-Packard Company certifies that this product met its published specifications at the time of shipment from the factory. Hewlett-Packard further certifies that its calibration measurements are traceable to the National Institute of Standards and Technologies, to the extent allowed by the the Institute's calibration facility, and to the calibration facilities of other International Standards Organization members.

WARRANTY

This Hewlett-Packard instrument product is warranted against defects in materials and workmanship for a period of one year from date of shipment. During the warranty period, Hewlett-Packard Company will, at its option, either repair or replace products which prove to be defective.

For warranty service or repair, this product must be returned to a service facility designated by -hp-. Buyer shall prepay shipping charges to -hp- and -hp- shall pay shipping charges to return the product to Buyer. However, Buyer shall pay all shipping charges, duties, and taxes for products returned to -hp- from another country.

Duration and conditions of warranty for this instrument may be superceded when the instrument is integrated into (becomes a part of) other -hp- instrument products.

Hewlett-Packard warrants that its software and firmware designated by -hp- for use with an instrument will execute its programming instructions when properly installed on that instrument. Hewlett-Packard does not warrant that the operation of the instrument, or software, or firmware will be uninterrupted or error free.

LIMITATION OF WARRANTY

The foregoing warranty shall not apply to defects resulting from improper or inadequate maintenance by Buyer, Buyer-supplied software or interfacing, unauthorized modification or misuse, operation outside of the environmental specifications for the product, or improper site preparation or maintenance.

NO OTHER WARRANTY IS EXPRESSED OR IMPLIED. HEWLETT-PACKARD SPECIFICALLY DISCLAIMS THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

EXCLUSIVE REMEDIES

THE REMEDIES PROVIDED HEREIN ARE BUYER'S SOLE AND EXCLUSIVE REMEDIES. HEWLETT-PACKARD SHALL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER BASED ON CONTRACT, TORT, OR ANY OTHER LEGAL THEORY.

ASSISTANCE

Product maintenance agreements and other customer assistance agreements are available for Hewlett-Packard products.

For any assistance, contact your nearest Hewlett-Packard Sales and Service Office. Addresses are provided at the back of this manual.

F

Table of Contents

Contents

Chapter 1
General Information

Chapter 1

General Information

Manual Contents

The HP 3245A Command Reference Manual contains an alphabetical description of the HP 3245A Universal Source command set, including HP 3245A commands and HP-IB commands, and lists error messages which apply to the instrument. This manual has two chapters. An overview of the chapters follows.

Chapter 1 - General Information

This chapter includes a functional command summary, an alphabetical command summary, and a list of error messages for the HP 3245A command set. It also describes the command reference page format.

Chapter 2 - Commands

This chapter contains an alphabetical description of all HP 3245A commands and the seven HP-IB commands which apply to the instrument. Each command entry describes the command, shows syntax and parameters, and provides an example program or program segment.

Command Reference Page Description

As shown in Figure 1-1, a command reference page includes the command keyword, command description, command syntax and parameters, remarks, and programming example(s). A description of each item follows.

Command Keyword

The command keyword listing shows the command and the associated query command (if any), such as **DUTY/DUTY?**. If there are two acceptable spellings for the keyword, the second spelling is included in parentheses, such as **DISP (DSP)**, where **DSP** is the alternate form of **DISP**.

Command Description

The command description (title) is shown in bold. When the description is for a command and associated query command, the title refers to both commands. For example, in the **DUTY/DUTY?** description, **Set/Read Duty Cycle** means that the **DUTY** command sets the duty cycle, while the **DUTY?** command reads the duty cycle.

Syntax

Shows the command keyword and applicable parameter(s) (Some commands have no parameters.) Parameters are shown in italics with optional parameters enclosed in square brackets ([]). For example, **APPLY WFV** *pp_amplitude* [,*array_name*] has required parameter *pp_amplitude* and optional parameter *array_name*.

Parameters

Describes parameter choices and ranges and includes power-on and default value (if any) for each parameter. The default value is used when a value is not specified for an optional parameter.

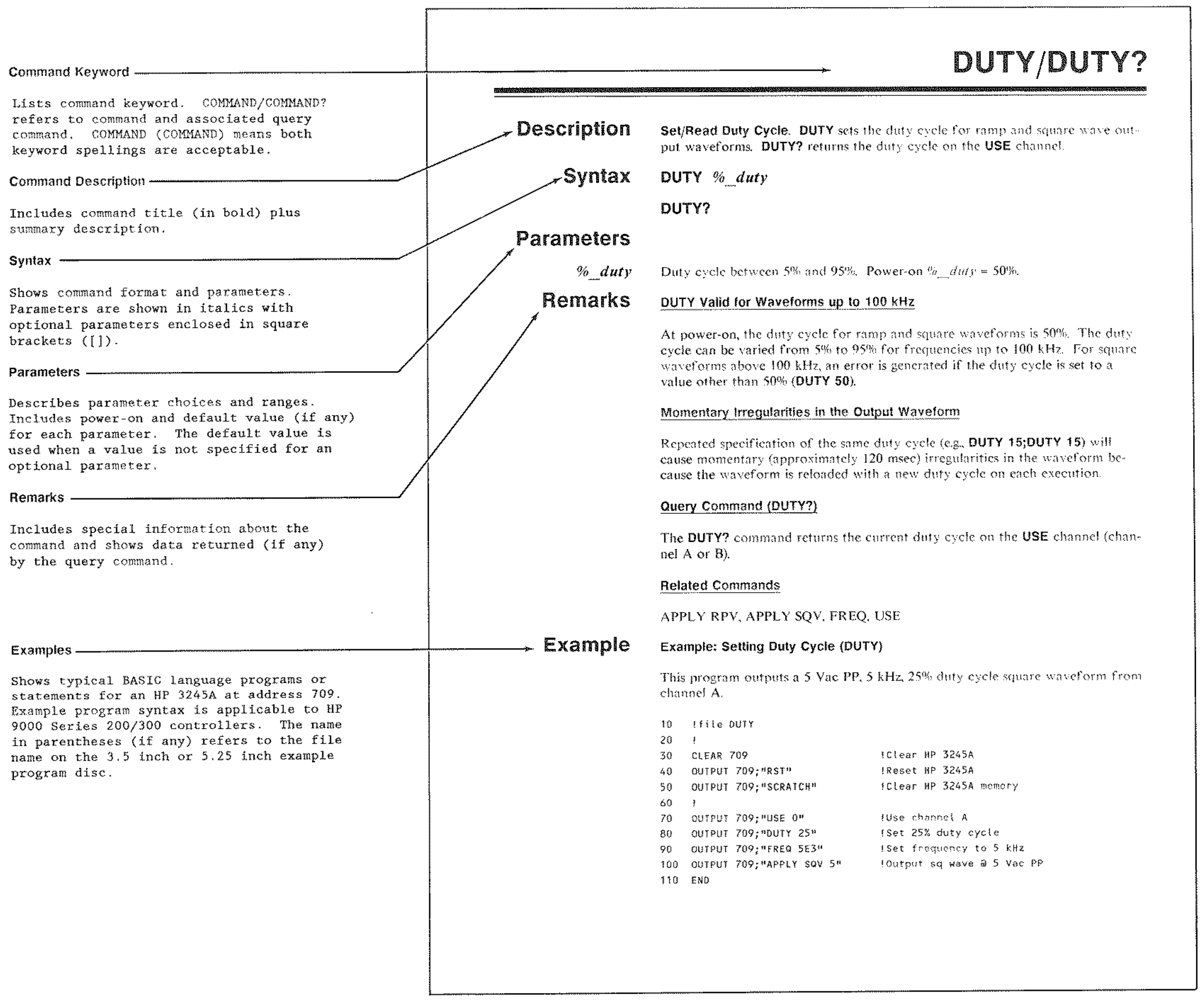
Remarks

Includes special information about the command and shows data returned (if any) by the query command. For example, **DUTY?** is the associated query command for the **DUTY** command.

Examples

The example section shows typical BASIC language programs or statements for an HP 3245A at address 709. The example program syntax is applicable to HP 9000 Series 200/300 controllers. The name in parentheses (if any) refers to the file name on the 3.5 inch or 5.25 inch example program disc.

For example, in the title **Example: Setting Duty Cycle (DUTY)**, the program listing is stored on the Example Programs disc titled "Example Programs - Command Reference" under file name DUTY. To run this program, load the appropriate disc in your controller, type LOAD "DUTY", and press the RUN key.



Command Keyword

Lists command keyword. COMMAND/COMMAND? refers to command and associated query command. COMMAND (COMMAND) means both keyword spellings are acceptable.

Command Description

Includes command title (in bold) plus summary description.

Syntax

Shows command format and parameters. Parameters are shown in italics with optional parameters enclosed in square brackets ([]).

Parameters

Describes parameter choices and ranges. Includes power-on and default value (if any) for each parameter. The default value is used when a value is not specified for an optional parameter.

Remarks

Includes special information about the command and shows data returned (if any) by the query command.

Examples

Shows typical BASIC language programs or statements for an HP 3245A at address 709. Example program syntax is applicable to HP 9000 Series 200/300 controllers. The name in parentheses (if any) refers to the file name on the 3.5 inch or 5.25 inch example program disc.

DUTY/DUTY?

Description

Set/Read Duty Cycle. **DUTY** sets the duty cycle for ramp and square wave output waveforms. **DUTY?** returns the duty cycle on the **USE** channel.

Syntax

DUTY *%_duty*

DUTY?

Parameters

%_duty Duty cycle between 5% and 95%. Power-on *%_duty* = 50%.

Remarks

DUTY Valid for Waveforms up to 100 kHz

At power-on, the duty cycle for ramp and square waveforms is 50%. The duty cycle can be varied from 5% to 95% for frequencies up to 100 kHz. For square waveforms above 100 kHz, an error is generated if the duty cycle is set to a value other than 50% (**DUTY 50**).

Momentary Irregularities in the Output Waveform

Repeated specification of the same duty cycle (e.g., **DUTY 15;DUTY 15**) will cause momentary (approximately 120 msec) irregularities in the waveform because the waveform is reloaded with a new duty cycle on each execution.

Query Command (DUTY?)

The **DUTY?** command returns the current duty cycle on the **USE** channel (channel A or B).

Related Commands

APPLY RPV, APPLY SQV, FREQ, USE

Example

Example: Setting Duty Cycle (DUTY)

This program outputs a 5 Vac PP, 5 kHz, 25% duty cycle square waveform from channel A.

```
10   !file DUTY
20   !
30   CLEAR 709                    !Clear HP 3245A
40   OUTPUT 709;"RST"             !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"         !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"           !Use channel A
80   OUTPUT 709;"DUTY 25"         !Set 25% duty cycle
90   OUTPUT 709;"FREQ 5E3"        !Set frequency to 5 kHz
100  OUTPUT 709;"APPLY SQV 5"     !Output sq wave @ 5 Vac PP
110  END
```

Figure 1-1. Command Reference Page Description

Alphabetical Command Summary

Table 1-1 is an alphabetical listing of HP 3245A and applicable HP-IB commands. In Table 1-1, a "Y" in the Menu column means that the command is a Menu command; a "Y" in the Keys column means that the command may be entered from the front panel; and a "Y" in the Rem column means that the command may be executed from the front panel when the HP 3245A is in remote.

Table 1-1. HP 3245A Alphabetical Command Summary

Command	Title	Menu	Keys	Rem
	--A--			
ABORT	Abort Subroutine	Y	N	N
ABORT (IFC)	Clear HP-IB Interface (HP-IB)	N	N	N
ABS	Absolute Value	N	N	N
ADDR?	Same as ADDRESS?	N	Y	Y
ADDRESS	Set HP-IB Address	Y	N	N
ADDRESS?	Read HP-IB address	Y	Y	Y
AND	Logical AND	N	N	N
APPLY?	Output Function Query	Y	N	Y
APPLY ACI	Apply Current Sine Wave	Y	Y	N
APPLY ACV	Apply Voltage Sine Wave	Y	Y	N
APPLY DCI	Apply DC Current	Y	Y	N
APPLY DCMEMI	Apply Triggered DCI	Y	Y	N
APPLY DCMEMV	Apply Triggered DCV	Y	Y	N
APPLY DCV	Apply DC Voltage	Y	Y	N
APPLY RPI	Apply Current Ramp Wave	Y	Y	N
APPLY RPV	Apply Voltage Ramp Wave	Y	Y	N
APPLY SQI	Apply Current Square Wave	Y	Y	N
APPLY SQV	Apply Voltage Square Wave	Y	Y	N
APPLY WFI	Apply Current Arbitrary Wave	Y	Y	N
APPLY WFV	Apply Voltage Arbitrary Wave	Y	Y	N
ARANGE	Set Autorange	Y	Y	N
ARANGE?	Read Autorange	Y	N	Y
ATN	Arctangent	N	N	N
	--B--			
BEEP	Beep	Y	N	Y
BINAND	Binary AND	N	N	N
BINCMP	Binary Complement	N	N	N
BINEOR	Binary Exclusive-OR	N	N	N
BINIOR	Binary Inclusive-OR	N	N	N
BIT	Read Bit Value	N	N	N
BLOCKOUT	Block Output Mode	Y	N	N

Table 1-1. HP 3245A Command Summary (cont'd)

Command	Title	Menu	Keys	Rem
	--C--			
CAL	Channel Calibration	Y	N	N
CALEN?	Read CAL Jumper Setting	Y	N	Y
CALL	Call Subroutine	Y	N	Y
CALNUM?	Calibration Number Query	Y	N	Y
CALSTR	Store Calibration Data	N	N	N
CALSTR?	Read Calibration Data	Y	N	Y
CAT	Catalog	Y	N	Y
CLEAR	HP-IB Device Clear (HP-IB)	N	N	N
CLR	Device Clear	Y	Y	N
CLROUT	Clear Output Buffer	Y	N	N
COMPRESS	Compress Subroutine	Y	N	N
CONT	Continue Subroutine	Y	N	N
COS	Cosine	N	N	N
CRESET	Channel Reset	Y	N	N
CTYPE	Channel Type Query	N	N	N
CTYPE?	Channel Type Query	Y	N	Y
	--D--			
DCOFF	Set DC Offset Voltage	Y	Y	N
DCOFF?	Read DC Offset Voltage	Y	Y	Y
DCRES	Set DC Resolution	Y	Y	N
DCRES?	Read DC Resolution	Y	Y	Y
DEFKEY	Define Key	N	Y	N
DEFKEY?	Read Defined Key	N	Y	Y
DELAY	Set Output Delay	Y	Y	N
DELAY?	Read Output Delay	Y	Y	Y
DELSUB	Delete Subroutine	Y	N	N
DEMO	Demonstration Waveforms	Y	N	N
DIM	Dimension REAL array	N	Y	N
DISP	Enable Front Panel Display	N	N	N
DISP?	Read Front Panel Display	N	N	N
DIV	Division	N	N	N
DRIVETB0	Set Trigger Bus 0 Source	Y	Y	N
DRIVETB0?	Read Trigger Bus 0 Source	Y	Y	Y
DRIVETB1	Set Trigger Bus 1 Source	Y	Y	N
DRIVETB1?	Read Trigger Bus 1 Source	Y	Y	Y
DSP	Same as DISP	N	N	N
DTEST	Data Test	Y	N	N
DTST	Same as DTEST	Y	N	N
DUTY	Set Duty Cycle	Y	Y	N
DUTY?	Read Duty Cycle	Y	Y	Y

Table 1-1. HP 3245A Command Summary (cont'd)

Command	Title	Menu	Keys	Rem
	--E--			
ECHO	Echo	N	N	N
END	End	N	N	N
END IF	End of IF...END IF Loop	N	N	N
END WHILE	End of WHILE..END WHILE Loop	N	N	N
ERR?	Error Query	N	N	N
ERRSTR?	Error String Query	Y	Y	Y
EXOR	Exclusive-OR	N	N	N
EXP	Exponent	N	N	N
	--F--			
FASTAMP	Fast Ampl/Offset Changes	Y	N	N
FASTFREQ	Fast Frequency Changes	Y	N	N
FETCH	Fetch Value	N	N	N
FILL	Fill Array	N	N	N
FILLAC	Fill Array W/Sine Wave	N	N	N
FILLBIN	Fill Array W/Binary	N	N	N
FILLRP	Fill Array W/Ramp Wave	N	N	N
FILLWF	Fill Array W/Arb Wave	N	N	N
FOR..NEXT	For-Next Loop	N	N	N
FREQ	Set Output Frequency	Y	Y	N
FREQ?	Read Output Frequency	Y	Y	Y
FTEST	Fixtured Self-Test	Y	N	N
FTST	Same as FTEST	Y	N	N
	--H--			
HELP	HELP Function	Y	Y	Y
	--I--			
ID?	Model Number Query	Y	N	Y
IDN?	Identity Query	Y	N	Y
IF..END IF	If-End If Branching	N	N	N
IMP	Set Output Impedance	Y	Y	N
IMP?	Read Output Impedance	Y	Y	Y
INBUF	Enable/Disable Input Buffer	Y	N	N
INTEGER	Dim INTEGER Array/Variable	N	N	N

Table 1-1. HP 3245A Command Summary (cont'd)

Command	Title	Menu	Keys	Rem
	--L--			
LCL	Same as LOCAL	N	N	N
LET	LET Assignment	N	N	N
LGT	Logarithm	N	N	N
LIST	List Subroutine	Y	N	Y
LOCAL	Go to Local	Y	Y	Y
LOCAL (GTL)	Go to Local (HP-IB)	N	N	N
LOCAL LOCKOUT	Local Lockout (HP-IB)	N	N	N
LOCK	Lock Front Panel Keyboard	N	N	N
LOG	Natural Logarithm	N	N	N
	--M--			
MEM	Memory Mode	N	N	N
MEMAVAIL?	Memory Available Query	Y	N	Y
MOD	Modulo	N	N	N
MON	Monitor Conditions	Y	N	Y
	--N--			
NOT	Inclusive-NOT	N	N	N
	--O--			
OFORMAT	Output Format	Y	N	N
OR	Inclusive-OR	N	N	N
OUTBUF	Output Buffer	Y	N	N
OUTPUT?	Output Value Query	Y	N	Y
	--P--			
PANG	Set Phase Angle	Y	Y	N
PANG?	Read Phase Angle	Y	Y	Y
PAUSE	Pause Subroutine	Y	N	N
PAUSED?	Pause Query	Y	N	Y
PHSYNC	Synchronized Output Mode	Y	Y	N
PONSRQ	Power-On SRQ	Y	N	N
	--R--			
RANGE	Set Output Range	Y	Y	N
RANGE?	Read Output Range	Y	Y	Y
READY?	Ready Query	Y	N	Y
REAL	Dim REAL Array/Variable	N	N	N
REFIN	Set Ref Frequency Input	Y	Y	N
REFIN?	Read Ref Frequency Input	Y	Y	Y

Table 1-1. HP 3245A Command Summary (cont'd)

Command	Title	Menu	Keys	Rem
	--R (cont'd) --			
REFOUT	Set Ref Frequency Output	Y	Y	N
REFOUT?	Read Ref Frequency Output	Y	Y	Y
REMOTE	Set REN Line TRUE (HP-IB)	N	N	N
RESET	Reset Instrument/Channel	Y	Y	N
RETURN	Return to Caller	N	N	N
REV?	Revision Query	Y	N	Y
RMT	Remote	Y	N	N
ROTATE	Rotate Bits	N	N	N
RQS	Set Service Request Enable	Y	N	N
RQS?	Read Service Request Enable	Y	N	Y
RST	Same as RESET	N	Y	N
RSTATE	Recall Stored State	Y	Y	N
RSTATEA	Recall Channel A State	Y	N	N
RSTATEB	Recall Channel B State	Y	N	N
RUN	Run Subroutine	Y	N	N
RUNNING?	Running Query	Y	N	Y
	--S--			
SCRATCH	Delete All	Y	N	N
SECURE	Calibration Security	Y	N	N
SER?	Serial Number Query	Y	N	Y
SET	Send HP 3245A State	Y	N	N
SET?	Read HP 3245A State	Y	N	N
SET TIME	Set Time	Y	N	N
SHIFT	Shift Bits	N	N	N
SIN	Sine	N	N	N
SIZE?	Size Query	Y	N	Y
SPOLL	Serial Poll (HP-IB)	N	N	N
SQR	Square Root	N	N	N
SRQ	Programmed Service Request	Y	N	Y
SSTATE	Store HP 3245A State	Y	Y	N
SSTATEA	Store Channel A State	Y	N	N
SSTATEB	Store Channel B State	Y	N	N
STA?	Status Word Query	Y	N	Y
STB?	Status Byte Query	Y	N	Y
STEP	Single Step Subroutine	Y	N	N
STOREATOB	Copy Channel A State	Y	N	N
STOREBTOA	Copy Channel B State	Y	N	N
SUB	Begin Subroutine	N	N	N
SUBEND	End Subroutine	N	N	N
SYNCOUT	Set Sync Out	Y	Y	N
SYNCOUT?	Read Sync Out	Y	Y	Y

Table 1-1. HP 3245A Command Summary (cont'd)

Command	Title	Menu	Keys	Rem
	--T--			
TB0?	Trigger Bus 0 Query	Y	N	Y
TB1?	Trigger Bus 1 Query	Y	N	Y
TERM	Set Output Terminal	Y	Y	N
TERM?	Read Output Terminal	Y	Y	Y
TEST	Confidence Test	Y	Y	N
TIME	Read Time	Y	N	Y
TRG	Pulse Trigger Bus	Y	N	N
TRIGFREQ	Triggered Frequency Mode	Y	N	N
TRIGGER	Group Execute Trigger (HP-IB)	N	N	N
TRIGIN	Set Trigger Input Source	Y	Y	N
TRIGIN?	Read Trigger Input Source	Y	Y	Y
TRIGMODE	Set Trigger Mode	Y	Y	N
TRIGMODE?	Read Trigger Mode	Y	Y	Y
TRIGOUT	Enable Trigger Output	Y	Y	N
TRIGOUT?	Read Trigger Output	Y	Y	Y
TST	Same as TEST	Y	Y	N
	--U--			
USE	Set Use Channel	N	Y	N
USE?	Read Use Channel	Y	N	Y
	--V--			
VREAD	Read Variable/Array	N	N	N
	--W--			
WAIT	Wait	Y	N	Y
WAITFOR	Wait For Event	N	N	N
WHILE	While-End While Loop	N	N	N

Functional Command Summary

Figure 1-2 summarizes HP 3245A command functions divided into four functional categories: Outputs/Waveforms; Instrument Functions; Arrays/Subroutines; and Input/Output Operations. A functional command summary follows Figure 1-2. In the summary, a "(?)" symbol indicates the command has an associated query command. For example, the ARANGE (?) entry means that ARANGE sets autorange mode, while ARANGE? reads the autorange mode.

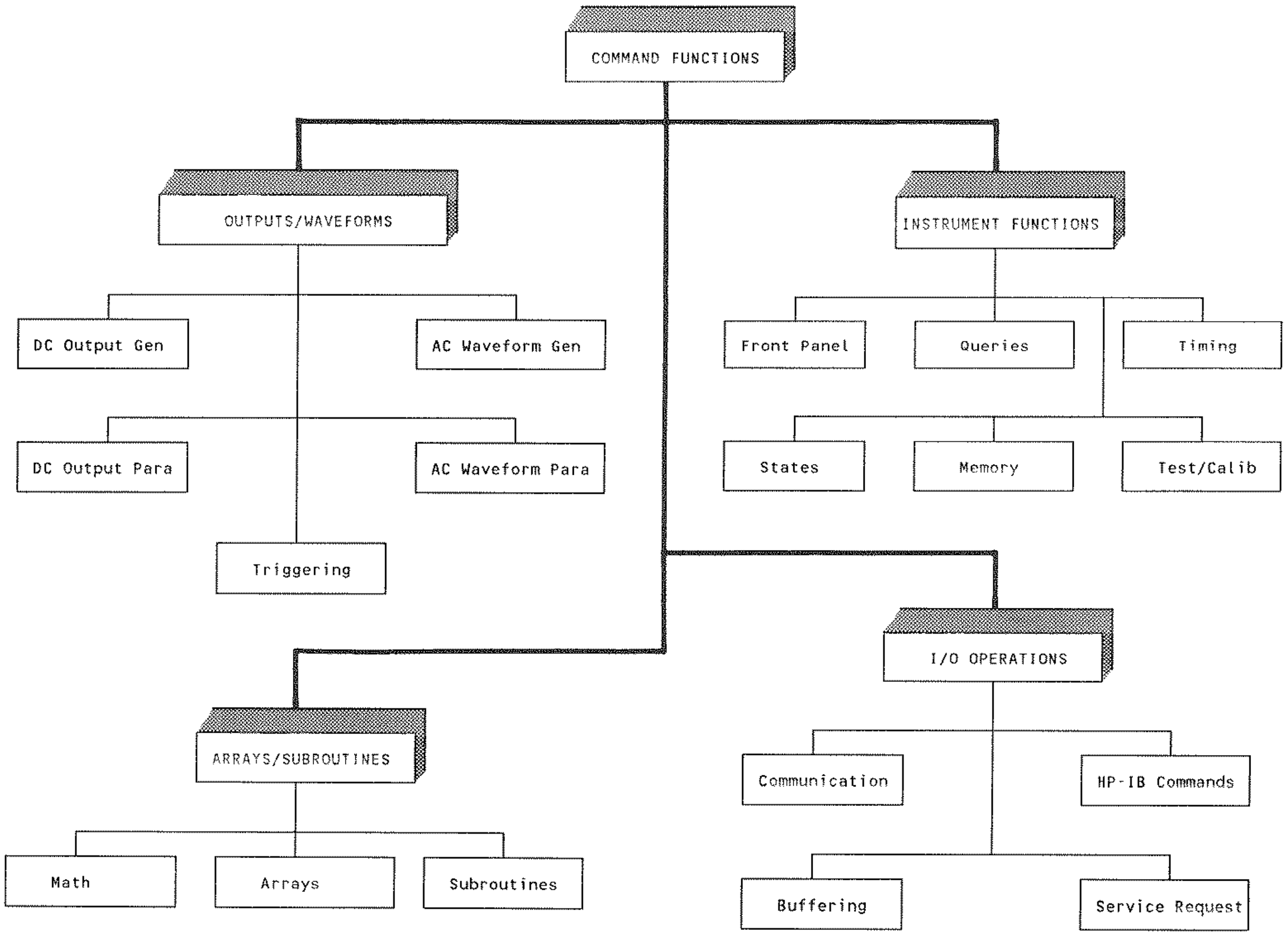


Figure 1-2. HP 3245A Commands by Function

OUTPUTS/WAVEFORMS

DC Output Generation

APPLY?	Output Function Query
APPLY DCI	Apply DC Current
APPLY DCMEMI	Apply Triggered DCI
APPLY DCMEMV	Apply Triggered DCV
APPLY DCV	Apply DC Voltage

DC Output Parameters

ARANGE (?)	Set/Read Autorange
DCRES (?)	Set/Read DC Resolution
DELAY (?)	Set/Read Output Delay
IMP (?)	Set/Read Output Impedance
OUTPUT?	Output Value Query
RANGE (?)	Set/Read Output Range
TERM (?)	Set/Read Output Terminal
USE (?)	Set/Read Use Channel

AC Waveform Parameters

ARANGE (?)	Set/Read Autorange
DCOFF (?)	Set/Read DC Offset Voltage
DELAY (?)	Set/Read Output Delay
DUTY (?)	Set/Read Duty Cycle
FREQ (?)	Set/Read Output Frequency
IMP (?)	Set/Read Output Impedance
OUTPUT?	Output Value Query
PANG (?)	Set/Read Phase Angle
RANGE (?)	Set/Read Output Range
TERM (?)	Set/Read Output Terminal
USE (?)	Set/Read Use Channel

AC Waveform Generation

APPLY?	Output Function Query
APPLY ACV	Apply Current Sine Wave
APPLY ACV	Apply Voltage Sine Wave
APPLY RPI	Apply Current Ramp Wave
APPLY RPV	Apply Voltage Ramp Wave
APPLY SQI	Apply Current Square Wave
APPLY SQV	Apply Voltage Square Wave
APPLY WFI	Apply Current Arbitrary Wave
APPLY WFV	Apply Voltage Arbitrary Wave
DEMO	Demonstration Waveforms
FASTAMP	Fast Ampl/Offset Changes
FASTFREQ	Fast Frequency Changes
TRIGFREQ	Triggered Frequency Mode

Triggering Commands

DRIVETBn (?)	Set/Read Trigger Bus n Source
PHSYNC	Synchronized Output Mode
REFIN (?)	Set/Read Ref Frequency Input
REFOUT (?)	Set/Read Ref Frequency Output
SYNCOUT (?)	Set/Read Sync Out Destination
TBn?	Trigger Bus n Query
TRG	Pulse Trigger Bus
TRIGIN (?)	Set/Read Trigger Input Source
TRIGMODE (?)	Set/Read Trigger Mode
TRIGOUT (?)	Enable/Read Trigger Output

INSTRUMENT FUNCTIONS

Front Panel Functions

DEFKEY (?)	Define Key/Read Defined Key
DISP (?)	Enable/Read Front Panel Display
MON	Monitor Conditions

General Query Functions

ADDRESS?	Read HP-IB Address
CTYPE?	Channel Type Query
ERR?	Error Query
ERRSTR?	Error String Query
ID?	Model Number Query
IDN?	Identity Query
MEMAVAIL?	Memory Available Query
REV?	Revision Query
SER?	Serial Number Query
SIZE?	Size Query
USE?	Read Use Channel

Timing Functions

SET TIME	Set Time
TIME	Read Time
WAIT	Wait
WAITFOR	Wait For Event

Storing/Recalling States

RSTATE	Recall Stored State
RSTATEA	Recall Channel A State
RSTATEB	Recall Channel B State
SET (?)	Send/Read HP 3245A State
SSTATE	Store HP 3245A State
SSTATEA	Store Channel A State
SSTATEB	Store Channel B State
STOREATOB	Copy Channel A State
STOREBTOA	Copy Channel B State

Memory Management

MEM	Set Memory Mode
MEMAVAIL?	Query Available Memory

Test/Calibration

CAL	Channel Calibration
CALEN?	Read CAL Jumper Setting
CALNUM?	Calibration Number Query
CALSTR (?)	Store/Read Calibration Data
DTEST	Data Self-Test
FTEST	Fixtured Self-Test
SECURE	Calibration Security
TEST	Confidence Self-Test

ARRAYS/SUBROUTINES

General Purpose Math

+	Addition
-	Subtraction
*	Multiplication
/	Division
^	Exponentiation
ABS	Absolute Value
AND	Logical AND
ATN	Arctangent
BINAND	Binary AND
BINCMP	Binary Complement
BINEOR	Binary Exclusive-OR
BINIOR	Binary Inclusive-OR
BIT	Read Bit Value
COS	Cosine
DIV	Division
EXOR	Exclusive-OR
EXP	Exponent
LGT	Logarithm
LOG	Natural Logarithm
MOD	Modulo
NOT	Inclusive-NOT
OR	Inclusive-OR
ROTATE	Rotate Bits
SHIFT	Shift Bits
SIN	Sine
SQR	Square Root

Variable/Array Operations

CAT	Catalog Arrays/Subroutines
DIM	Dimension REAL Array
FETCH	Fetch Value
FILL	Fill Array
FILLAC	Fill Array w/Sine Wave
FILLBIN	Fill Array w/Binary
FILLRP	Fill Array w/Ramp Wave
FILLWF	Fill Array w/Arb Wave
INTEGER	Dim INTEGER Array/Variable
LET	LET Assignment
MEM	Memory Mode
SCRATCH	Delete All
VREAD	Read Variable/Array

Subroutine Commands

ABORT	Abort Subroutine
CALL	Call Subroutine
CAT	Catalog Arrays/Subroutines
COMPRESS	Compress Subroutine
CONT	Continue Subroutine
DELSUB	Delete Subroutine
END IF	End of If-End If Loop
END WHILE	End of While-End While Loop
FOR..NEXT	For-Next Loop
IF..END IF	If-End If Branching
LIST	List Subroutine
PAUSE	Pause Subroutine
PAUSED?	Pause Query
RETURN	Return to Caller
RUN	Run Subroutine
RUNNING?	Running Query
SCRATCH	Delete All
STEP	Single Step Subroutine
SUB	Begin Subroutine
SUBEND	End Subroutine
WHILE	While-End While Loop

INPUT/OUTPUT OPERATIONS

HP-IB Communication

ADDRESS (?)	Set/Read HP-IB Address
BEEP	Enable Beep Mode
CLR	Device Clear
ECHO	Echo
LOCAL	Go to Local
LOCK	Lock Front Panel Keyboard
RESET	Reset Instr/Channel
REM	Remote
TEST	Confidence Self-Test
USE (?)	Set/Read Use Channel

Input/Output Buffering

BLOCKOUT	Block Output Mode
CLROUT	Clear Output Buffer
END	End or Identify
INBUF	Enable/Disable Input Buffer
MEM	Memory Mode
OFORMAT	Output Format
OUTBUF	Enable Output Buffer
READY?	Query Ready Status
VREAD	Read Data From Memory

Interrupts/Service Requests

PONSRQ	Power-on SRQ
RQS (?)	Set/Read Service Request Enable
SRQ	Programmed Service Request
STA?	Status Word Query
STB?	Status Byte Query

HP-IB Commands

ABORT	Clear HP-IB Interface
CLEAR	Device Clear/Sel Dev Clear
LOCAL	Go to Local
LOCAL LOCKOUT	Local Lockout
REMOTE	Set REN Line TRUE
SPOLL	Serial Poll
TRIGGER	Group Execute Trigger

Error Messages

HP 3245A error messages follow.

HP 3245A Error Messages

No.	Message	Meaning
Syntax Errors		
0	"NO ERROR"	No errors occurred.
1	"INCOMPLETE COMMAND"	Command is missing one or more parameters.
2	"SYNTAX"	Command keyword and/or parameters are incorrect.
3	"CANNOT RE-TYPE A VARIABLE"	Attempt to redefine a REAL variable as an INTEGER variable or vice-versa.
4	"ERROR IN #A BLOCK"	The block of data from SET? has changed or is no longer in a form SET can use.
5	"ARRAY SIZE OR TYPE MISMATCH"	REAL used, INTEGER expected or vice-versa.
6	"COMMAND TOO LONG"	Command exceeds maximum number of characters allowe.
Subroutine Operations		
21	"NOT ALLOWED IN SUB"	Command not allowed in HP 3245A subroutine.
22	"ALLOWED ONLY IN SUB"	Command allowed ONLY in HP 3245A subroutine.
23	"NEXT WITHOUT FOR"	No matching FOR in FOR.. NEXT loop.
24	"NEXT VARIABLE NOT SAME AS FOR VARIABLE"	FOR..NEXT variables do not match.
25	"EXPECTED NEXT"	No NEXT statement in a FOR..NEXT loop.

26	"ELSE OR END IF WITHOUT IF"	No IF statement in IF..END IF branching
27	"EXPECTED END IF"	No END IF statement in IF..END IF branching.
28	"END WHILE WITHOUT WHILE"	No WHILE statement in WHILE..END WHILE loop.
29	"EXPECTED END WHILE"	No END WHILE statement in WHILE..END WHILE loop.
30	"CONTROL VARIABLE IN USE"	Nested loops have same control variable.

Memory Operations

41	"OUT OF MEMORY"	Out of subroutine or variable space.
42	"TOO MANY NESTED CALLS"	Exceeded maximum number of nested calls.
43	"TOO MANY NESTED STRUCTURES"	Exceeded maximum number of nested subroutines.

Subroutine Execution/Stored States

51	"SUB WAS DELETED"	A call to deleted sub A occurs within running subroutine B.
52	"NO ACTIVE SUB"	Subroutine is not defined or is paused.
53	"SUB NOT PAUSED"	Subroutine is not in paused state.
54	"SUB IS RUNNING"	Subroutine is running. Cannot execute command.
55	"CANNOT DELSUB AN ACTIVE SUB"	Attempt to delete an active subroutine.
56	"RSTATE"	See SSTATE and the state memory specified is empty. No state was stored.
57	"SSTATE"	Number of state was out of range. Range for channel A only is 0 - 13. Range for channel A and B is 0 -6.
58	"SUB WAS COMPRESSED"	Cannot list a compressed subroutine.

Command Parameters

61	"OUT OF RANGE"	A value outside expected range was entered.
62	"CHANNEL MISSING"	Command for channel B was entered without channel B installed.
64	"UNSUPPORTED COMMAND"	The source channel specified was not found. May indicate a hardware error.
65	"COMMAND INCOMPATIBLE WITH SETUP"	Command cannot be executed, given the present instrument configuration.
66	"SUBSCRIPT OUT OF BOUNDS"	Array subscript out of expected range (0 to 2047).
67	"ARRAY TOO SMALL"	Array size too small for all specified data to be entered into the array.
69	"MUST BE IN LOCAL"	When in remote mode, commands which alter instrument configuration cannot be executed.
70	"SETTINGS CONFLICT"	Duty cycle and frequency settings are incompatible.

General Errors

82	"BUSY TOO LONG"	03245-66501 PC board problem. (same as error 207).
83	"BP ERROR FROM SLOT"	03245-66501 PC board problem.

System Errors

91	"SYSTEM ERROR"	
92	"BUS ERROR"	03245-66501 PC board did not respond.
93	"SYSTEM TRAP"	
94	"MATH ERROR"	Math error such as attempt to divide by zero.
95	"CPU EXCEPTION"	

Calibration Setup

116	"CAL INPUT OUT OF RANGE"	Value was out of range or 03245-66501 PC board problem.

Channel Errors

161	"MIN/MAX LIMIT ERROR"	Array element out of range.
162	"EXCESSIVE TERMINAL VOLTAGE, RELAYS OPEN"	Refer to HP 3245A Operating and Programming Manual note (page 10-9).
163	"ENABLE CAL RAM"	J1(S1) on 03245-66501 PC board must be enabled.
164	"CAL RAM FAIL"	Checksum error. Recalibrate channel.
165	"SPURIOUS HW ERROR, RELAYS OPEN"	Reset HP 3245A and continue.
166	"ERROR STUCK, CARD"	A channel may be bad. Run TEST or FTEST.

Self-Test Errors

201	"PON TEST"	03245-66505, 03245-66506, or cable problem.
202	"DTACK FAIL"	03245-66501 problem.
203	"CARD ID FAIL"	03245-66501 problem.
204	"CARD BUSY (D7) STUCK HIGH"	03245-66501 problem.
205	"CARD BUSY (D7) STUCK LOW"	03245-66501 problem.
206	"BUSY TOO SHORT"	03245-66501 problem.
207	"BUSY TOO LONG"	03245-66501 problem.
208	"RE-TRIG BUSY TIME FAIL"	03245-66501 problem.
209	"BP/BUSY STUCK LOW"	03245-66501 problem.
211	"TRIGGER BUS FAIL"	03245-66501 or 03245-66502 problem. Run FTEST.
226	"SOURCE SELF TEST"	03245-66501 problem.
229	"REQUIRES CHAN A & B"	Option 001 required.

Contents

Chapter 2
Commands

Chapter 2
Commands

Introduction

This chapter contains an alphabetical listing of all HP 3245A commands and the seven HP-IB commands which apply to the HP 3245A.

Command Listing

An alphabetical description of HP 3245A and applicable HP-IB commands follows.

ABORT

Description

Abort Subroutine. Halts execution of any RUN subroutine and returns control to the controller or to the front panel.

Syntax

ABORT

Parameters

None.

Remarks

CALL Subroutines Cannot Be Aborted

When a subroutine is called (with **CALL**), input to the HP 3245A from the front panel or from HP-IB is held off until the subroutine completes. Thus, a CALL subroutine cannot be aborted by a subsequent **ABORT** command. **ABORT** within a CALL subroutine is ignored. **ABORT** executed during (and outside) a CALL subroutine is held off until the subroutine completes. To abort a CALL subroutine, press the front panel **Clear** key or send an HP-IB Device Clear statement.

ABORT Timeout Error

ABORT halts execution of a RUN subroutine following execution of the current command. If a subroutine command takes a long time (e.g., **WAIT 10**), **ABORT** times out and generates a "DEVICE CLEAR TO STOP" error message.

Do Not Use ABORT Inside a Subroutine

ABORT executed within a RUN subroutine suspends subroutine execution and then generates "ERR 54: SUB IS RUNNING". **ABORT** executed during (and outside) a RUN subroutine aborts the subroutine (and any nested subroutines) at the end of the currently executing command.

Related Commands

RUN, SUB, SUBEND

Example

Example: Aborting a Subroutine (ABORT)

This program executes RUN subroutine SUBABORT. When the program executes, "SUBROUTINE RUNNING" and a number from 0 to 9 are alternately displayed on the 3245A. To abort this subroutine, place the 3245A in "local" and execute **ABORT** from the front panel. You can also abort a subroutine over HP-IB (`OUTPUT 709;"ABORT"`). If aborted from the front panel, the display returns to the power-on state. If aborted over HP-IB, the display will show either a number in the count loop or "SUBROUTINE RUNNING", depending on when the abort occurs.

```
10  !file ABORT
20  !
30  CLEAR 709                                       !Clear HP 3245A
40  OUTPUT 709;"RST"                                !Reset HP 3245A
50  OUTPUT 709;"SCRATCH"                            !Clear HP 3245A memory
60  !
70  OUTPUT 709;"SUB SUBABORT"                       !Begin subroutine
80  OUTPUT 709;"  FOR I = 0 TO 9"                   !Begin count loop
90  OUTPUT 709;"    DISP 'SUBROUTINE RUNNING'"      !Disp message
100 OUTPUT 709;"    WAIT 1"                         !Wait one second
110 OUTPUT 709;"    DISP I"                         !Disp count (0 to 9)
120 OUTPUT 709;"    WAIT 1"                         !Wait one second
130 OUTPUT 709;"  NEXT I"                           !Increment count
140 OUTPUT 709;"SUBEND"                             !End subroutine
150 !
160 OUTPUT 709;"RUN SUBABORT"                       !Run subroutine
170 END
```

ABORT (IFC)

Description Clears the HP 3245A interface circuitry and aborts all HP-IB communication. The 3245A becomes unaddressed, however; it remains in the remote state and no other instrument configuration is changed.

Syntax **ABORT**

Example

```
ABORT 7     !Clears HP 3245A interface circuitry.
```

Description

Absolute Value. Returns the absolute value of the specified argument.

Syntax

ABS (*argument*)

Parameters

argument

The *argument* parameter must be a number or numeric expression (enclosed in parentheses).

Remarks

None.

Example

Example: Using the ABS Command (ABS)

This program uses **ABS** to determine the absolute value of (10/-2) and displays the result (5) on the controller CRT.

```
10   !file ABS
20   !
30   CLEAR 709                        !Clear HP 3245A
40   OUTPUT 709;"RST"                 !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"             !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"               !Use channel A
80   OUTPUT 709;"  VREAD ABS(10/-2)"  !Read absolute value
90   ENTER 709;A                      !Enter result
100  PRINT "Absolute Value = ";A      !Display result
110  END
```

A typical return is:

```
Absolute Value = 5
```

ADDRESS/ADDRESS? (ADDR?)

Description

Set/Read HP-IB Address. **ADDRESS** sets the HP-IB address of the HP 3245A. **ADDRESS?** (or **ADDR?**) returns the current HP-IB address of the HP 3245A.

Syntax

ADDRESS *address*

ADDRESS?

Parameters

address Sets the HP-IB address. Valid range is 0 to 30.

Remarks

Setting HP-IB Address From Front Panel

To set the HP-IB address from the HP 3245A front panel, scroll to the **ADDRESS** command and then press the required numeric keys for the address to be set. Then, press the Enter key to enter the new address.

Data Destination (ADDRESS?)

If the **MEM** command is used, the value returned by **ADDRESS?** is stored in the specified variable or array. When **ADDRESS?** is executed from the front panel, the response is displayed on the front panel display. When **ADDRESS?** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

HP-IB Data Format (ADDRESS?)

The HP 3245A returns numeric results in either ASCII or binary format (see the **OFORMAT** command). In the default ASCII format, **ADDRESS?** returns integer results in 6-digit signed notation. In the binary format, **ADDRESS?** returns integer results in 16-bit, 2's complement notation.

Query Command (ADDRESS?)

The **ADDRESS?** (or **ADDR?**) command returns a one-digit or two-digit HP-IB address. Valid addresses for the HP 3245A are 0 through 30. The HP 3245A is shipped from the factory with an HP-IB address of 09.

Related Commands

None.

Example

Example: Setting/Reading HP-IB Address

To set the HP 3245A HP-IB address to 17 (bus address 717) from the front panel, use the MENU keys to display the **ADDRESS** command. Then, press the **1**, the **7**, and the **Enter** keys to set the new address. To read the new address from the front panel, press the **Enter** and then the **Address** key on the front panel keyboard (17 will be displayed).

Description

Logical AND. Returns a "0" or "1" based upon the logical AND of the specified arguments.

Syntax

argument **AND** *argument*

Parameters

argument

Each *argument* parameter may be a number, numeric variable, math function, array element, or numeric expression (enclosed in parentheses).

Remarks

AND Operation

The **AND** command returns "0" or "1" as shown in the following truth table. Any non-zero value (positive or negative) is treated as a logic "1". Only zero is treated as a logic "0".

A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

Related Commands

BINAND, OR

Example

Example: AND Statement in IF...END IF Loop (AND)

This program uses **AND** within an **IF...END IF** loop. In line 100, the value of "A AND B" is evaluated. If both A and B are NOT equal to zero, "TRUE - A AND B NON-ZERO" is displayed. If either, or both, A and B are equal to zero, "FALSE - A AND/OR B IS ZERO" is displayed.

The first time the subroutine is called, A = 3 and B = 2. The result of the AND operation is "TRUE" since A and B are both non-zero. The second time the subroutine is called, B = 0, so the result of the AND operation is "FALSE".

```
10   !file AND
20   !
30   CLEAR 709                         !Clear HP 3245A
40   OUTPUT 709;"RST"                  !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"              !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"                !Use channel A
80   OUTPUT 709;"SUB ANDLOOP"          !Begin subroutine
90   OUTPUT 709;"  INTEGER A,B"        !Define INTEGER variables
100  OUTPUT 709;"  IF A AND B THEN"    !Begin IF..END IF loop
110  OUTPUT 709;"    DISP 'TRUE - A AND B NON-ZERO'"
120  OUTPUT 709;"  ELSE"
340  OUTPUT 709;"    DISP 'FALSE - A AND/OR B IS ZERO'"
140  OUTPUT 709;"  END IF"             !End IF...END IF loop
```

```
150  OUTPUT 709;"SUBEND"                   !End subroutine
160  !
170  OUTPUT 709;"A=3;B=2"                  !Define A and B
180  OUTPUT 709;"CALL ANDLOOP"             !Call subroutine
190  WAIT 3
200  OUTPUT 709;"B=0"                      !Redefine B
210  OUTPUT 709;"CALL ANDLOOP"             !Call subroutine
220  END
```

Description

Output Function Query. Returns the last programmed output function from the **USE** channel.

Syntax

APPLY?

Parameters

None.

Remarks

Data Returned

The **APPLY?** command returns the last programmed output function (ACI, ACV, DCI, DCV, DCMEMI, DCMEMV, RPI, RPV, SQI, SQV, WFI, or WFV) from the **USE** channel.

Data Destination

When **APPLY?** is executed from the front panel, the response is displayed on the front panel display. When **APPLY?** is executed from the controller, the response is sent to the output buffer in the default ASCII format. If **MEM ON** is set, the value returned by **APPLY?** is stored in the specified variable or array.

Related Commands

APPLYs, USE

Example

Example: Reading Output Function (APPLYQ)

This program returns the last programmed output function (RPV) from channel A.

```
10   !file APPLYQ
20   !
30   CLEAR 709                      !Clear HP 3245A
40   OUTPUT 709;RST"                !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"           !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"             !Use channel A
80   OUTPUT 709;"APPLY RPV 1.0"     !Output ramp waveform
90   OUTPUT 709;"APPLY?"            !Query ch A output function
100  ENTER 709;A$                   !Enter result
110  PRINT A$                       !Display result
120  END
```

A typical return is:

```
RPV
```

APPLY ACI

Description

Apply Current Sine Waveform. APPLY ACI outputs an AC current sine waveform from the **USE** channel.

Syntax

APPLY ACI *pp_amplitude*

Parameters

pp_amplitude Peak-to-peak current amplitude between 0.00001 A and 0.2 Amps.

Remarks

AC Current Ranges

AC current waveforms can be generated on one of four ranges in the 0Ω or 50Ω output impedance mode. The following table shows the AC current ranges and maximum programmed output values available in each output impedance mode. You can set the output amplitude from 5% to 100% of the specified range.

Range	Maximum Programmed Output	Resolution
0.0002 A	0.0002 A	50 nA
0.002 A	0.002 A	500 nA
0.02 A	0.02 A	5 µA
0.2 A	0.2 A	50 µA

The Autorange Function (ARANGE)

When the autorange function is enabled (**ARANGE ON**), the HP 3245A automatically selects the lowest voltage or current range which will provide maximum accuracy for the desired output level. An error is generated if you attempt to set the output level greater than 100% of the highest voltage or current range available. For AC waveforms, an error is also generated if you attempt to set the peak-to-peak amplitude less than 5% of the lowest peak-to-peak range.

When the autorange function is disabled (**ARANGE OFF**), the instrument retains the present range regardless of the programmed output value. If the programmed value attempts to exceed 100% of the present range, an error is generated. For AC waveforms, an error is also generated if you attempt to set the peak-to-peak amplitude less than 5% of the present peak-to-peak range.

Selecting the Output Frequency (FREQ)

At power-on, the output frequency for AC sine waveforms is 1000 Hz. The output frequency can be varied from 0 to 1 MHz with 0.001 Hz resolution. Refer to the **FREQ** command for details on setting the output frequency.

Setting the DC Offset Voltage (DCOFF)

At power-on, the DC offset voltage for AC sine waveforms is 0 volts. The offset can be changed with the **DCOFF** command. The offset can be a positive or negative number such that the peak AC value of the output waveform, plus the magnitude of the offset, does not exceed 50% of the selected peak-to-peak range. Refer to the **DCOFF** command for details on selecting the DC offset voltage.

RMS to PP Conversion

APPLY ACI generates an AC current waveform with peak-to-peak (PP) values set by *pp_amplitude*. For outputs measured with an RMS multimeter, use $I_{PP} = 2.828*I_{RMS}$ to convert an RMS reading to the equivalent PP value.

Related Commands

DCOFF, FREQ, IMP, OUTPUT?, TERM, USE

Example

Example: Sine Waveform Current Output (APPL_ACI)

This program outputs a 0.005 Amp PP (5 mA) sine waveform from channel A.

```
10   !file APPL_ACI
20   !
30   CLEAR 709                       !Clear HP 3245A
40   OUTPUT 709;"RST"                !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"            !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"              !Use channel A
80   OUTPUT 709;"APPLY ACI 0.005"    !Output 5 mA PP sine waveform
90   END
```

APPLY ACV

Description

Apply Voltage Sine Waveform. APPLY ACV outputs an AC voltage sine waveform from the **USE** channel.

Syntax

APPLY ACV *pp_amplitude*

Parameters

pp_amplitude Peak-to-peak amplitude between 0.03125 Vac (PP) and 20 Vac (PP) in the 0Ω output impedance mode or between 0.015625 Vac (PP) and 10 Vac (PP) in the 50Ω mode (when terminated with a 50Ω load).

Remarks

AC Voltage Ranges

AC voltage waveforms can be generated on one of seven ranges in the 0Ω or 50Ω output impedance mode. The following table shows the AC voltage ranges and maximum programmed output values available in each output impedance mode. You can set the output amplitude from 10% to 100% of the specified range.

Output Impedance	Range (Volts P-P)	Maximum Programmed Output (Volts P-P)	Resolution
0Ω	0.3125V	0.3125V	156 µV
0Ω	0.625V	0.625V	312 µV
0Ω	1.25V	1.25V	625 µV
0Ω	2.5V	2.5V	1.25 mV
0Ω	5V	5V	2.5 mV
0Ω	10V	10V	5 mV
0Ω	20V	20V	10 mV
50Ω	0.15625V	0.15625V	78 µV
50Ω	0.3125V	0.3125V	156 µV
50Ω	0.625V	0.625V	312 µV
50Ω	1.25V	1.25V	625 µV
50Ω	2.5V	2.5V	1.25 mV
50Ω	5V	5V	2.5 mV
50Ω	10V	10V	5 mV

Setting the Output Impedance (IMP)

The output impedance for either channel can be set to 0Ω or 50Ω. In the 0Ω (power-on) mode, the voltage output is the same as the programmed output voltage with or without termination. However, in the 50Ω mode, the actual output voltage is equal to the programmed output voltage ONLY when the output is terminated with a 50Ω load. Other impedances will cause the output voltage to vary. Refer to the **IMP** command for details on setting the output impedance.

The Autorange Function (ARANGE)

When the autorange function is enabled (**ARANGE ON**), the HP 3245A automatically selects the lowest voltage or current range which will provide maximum accuracy for the desired output level. An error is generated if you attempt to set the output level greater than 100% of the highest voltage or current range available. For AC waveforms, an error is also generated if you attempt to set the peak-to-peak amplitude less than 10% of the lowest peak-to-peak range.

When the autorange function is disabled (**ARANGE OFF**), the instrument retains the present range regardless of the programmed output value. If the programmed value attempts to exceed 100% of the present range, an error is generated. For AC waveforms, an error is also generated if you attempt to set the peak-to-peak amplitude less than 10% of the present peak-to-peak range.

Selecting the Output Frequency (FREQ)

At power-on, the output frequency for AC waveforms is 1000 Hz. The output frequency can be varied from 0 to 1 MHz with 0.001 Hz resolution. Refer to the **FREQ** command for details on setting the output frequency.

Setting the DC Offset Voltage (DCOFF)

At power-on, the DC offset voltage for AC waveforms is 0 volts. The offset voltage can be changed using the **DCOFF** command. The offset can be a positive or negative number such that the peak AC value of the output waveform, plus the magnitude of the offset, does not exceed 50% of the selected peak-to-peak range. Refer to the **DCOFF** command for details on selecting the DC offset voltage.

PP to RMS Conversion

APPLY ACV generates an AC voltage waveform with peak-to-peak (PP) values set by *pp_amplitude*. For sine wave outputs measured with an RMS multimeter, use $V_{PP} = 2.828 * V_{RMS}$ to convert an RMS reading to an equivalent PP value.

Related Commands

DCOFF, FREQ, IMP, OUTPUT?, TERM, USE

Example

Example: Sine Waveform Voltage Output (APPL_ACV)

This program outputs a 4.5 Vac (PP) sine waveform from channel A.

```
10  !file APPL_ACV
20  !
30  CLEAR 709                    !Clear HP 3245A
40  OUTPUT 709;"RST"             !Reset HP 3245A
50  OUTPUT 709;"SCRATCH"         !Clear HP 3245A memory
60  !
70  OUTPUT 709;"USE 0"           !Use channel A
80  OUTPUT 709;"APPLY ACV 4.5"   !Output 4.5 Vac PP sine waveform
90  END
```

APPLY DCI

Description

Apply DC Current Output. APPLY DCI outputs a DC current from the **USE** channel.

Syntax

APPLY DCI *amps*

Parameters

amps Output Current. Valid range is -0.1 Amps DC to +0.1 Amps DC.

Remarks

DC Current Ranges

DC current outputs can be generated on one of four ranges in the high-resolution or low-resolution mode (refer to the **DCRES** command for details on resolution modes). The following table shows DC current ranges and maximum programmed output values.

Range	Maximum Programmed Output	Resolution High-Resolution	Resolution Low-Resolution
0.0001 A	0.0001 A	0.1 nA	50 nA
0.001 A	0.001 A	1 nA	500 nA
0.01 A	0.01 A	10 nA	5 μA
0.1 A	0.1 A	100 nA	50 μA

The Autorange Function (ARANGE)

When the autorange function is enabled (**ARANGE ON**), the lowest voltage or current range is selected which will provide maximum accuracy for the desired output level. An error is generated if an attempt is made to set the output level greater than 100% of the highest voltage or current range available.

When the autorange function is disabled (**ARANGE OFF**), the channel retains the present range regardless of the programmed output value. If the programmed value attempts to exceed 100% of the present range, an error is generated.

Related Commands

ARANGE, DCRES, OUTPUT?, TERM, USE

Example

Example: Generate DC Current Output (APPL_DCI)

This program outputs a DC current of 3.25 mA from channel A.

```
10   !file APPL_DCI
20   !
30   CLEAR 709                          !Clear HP 3245A
40   OUTPUT 709;"RST"                   !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"               !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"                 !Use channel A
80   OUTPUT 709;"  APPLY DCI 3.25E-3"   !Output 3.25 mA
90   END
```

Description

Apply Triggered DC Current. Outputs a triggered sequence of low-resolution current values from the **USE** channel.

Syntax

APPLY DCMEMI *length, array_name*

Parameters

length

Number of DC current values to be output. Valid range is 2 through 2048 values.

array_name

Name of array containing the DC current values. Valid range for each array element is -0.1 Amps DC to +0.1 Amps DC. **APPLY DCMEMI** generates DC currents using the low-resolution mode (see the **DCRES** command).

Array names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore character ("_"), or the question mark ("?"). Array names must not be the same as HP 3245A commands or parameters or stored state names.

Remarks

APPLY DCMEMI Operation

When **APPLY DCMEMI** is executed, the value in the first element of *array_name* is transferred to the channel and the equivalent current level is output. Note that a trigger is NOT required to output the first current level.

The remaining current levels are then output, one at a time, as high-to-low triggers are received from the trigger source selected by **TRIGIN**. After the last current level in the array is output, the next trigger causes wrap-around and the first current level is again output.

Using APPLY DCMEMI With Autorange (ARANGE)

If autorange is enabled (**ARANGE ON** or **RANGE AUTO**), **APPLY DCMEMI** searches *array_name* elements and selects the lowest current range which will provide maximum accuracy for the largest value in the array. If autorange is disabled, the instrument retains the range selected by the **RANGE** command and generates an error if any value in *array_name* attempts to exceed the present range.

Low-Resolution Mode is Used (DCRES)

APPLY DCMEMI automatically generates DC currents using the low-resolution mode (**DCRES LOW**). If the high-resolution mode (**DCRES HIGH**) is in effect when **APPLY DCMEMI** is selected, the channel temporarily goes to the low-resolution mode. When **APPLY DCMEMI** completes, the channel returns to the high-resolution mode.

Triggering the Channel (TRIGIN)

APPLY DCMEMI outputs DC current values in response to triggers received. **TRIGIN** selects the trigger source which activates the **APPLY DCMEMI** command. Refer to the **TRIGIN** command for details.

Related Commands

APPLY DCMEMV, DCRES, OUTPUT?, TERM, TRIGIN, USE

Example

Example: Triggered-Sequence DC Current Outputs (APP_DCMI)

This program outputs a triggered DC current sequence from channel A consisting of four current values: 0.01, 0.02, 0.03, and +0.04 Amps DC. When the program executes, **APPLY DCMEMI** outputs the first value in the array (0.01 Amps DC). The remaining currents are output, one at a time, at 3 second intervals when **TRIGIN SGL** (a software trigger) is executed. Note that the first output level (0.01 Amps) is output without a trigger.

```
10   !file APP_DCMI
20   !
30   CLEAR 709                          !Clear HP 3245A
40   OUTPUT 709;"RST"                   !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"               !Clear HP 3245A memory
60   !
70   OUTPUT 709;"DIM IOUT(3)"           !Dimension array
80   OUTPUT 709;"FILL IOUT 0.010,0.020,0.030,0.040"    !Enter array values
90   !
100  OUTPUT 709;"SUB I_OUTPUT"          !Begin subroutine
110  OUTPUT 709;"  USE 0"               !Use channel A
120  OUTPUT 709;"    APPLY DCMEMI 4,IOUT"    !Output triggered DCI
130  OUTPUT 709;"  WAIT 3"              !Wait 3 seconds
140  OUTPUT 709;"  FOR I = 1 TO 3"      !Begin count loop
150  OUTPUT 709;"    TRIGIN SGL"        !Single trigger
160  OUTPUT 709;"    WAIT 3"            !Wait 3 seconds
170  OUTPUT 709;"  NEXT I"              !Increment loop
180  OUTPUT 709;"SUBEND"                !End subroutine
190  !
200  OUTPUT 709;"RUN I_OUTPUT"          !Run subroutine
210  END
```

Description

Apply Triggered DC Voltage Output. Outputs a triggered sequence of low-resolution voltages from the **USE** channel.

Syntax

APPLY DCMEMV *length, array_name*

Parameters

length

Number of DC voltage values to be output. Valid range is 2 through 2048 values.

array_name

Name of array containing the DC voltage values. Valid range for each array element is -10 VDC to +10 VDC in the 0Ω mode or -5 VDC to +5 VDC in the 50 Ω mode. **APPLY DCMEMV** generates DC voltages using the low-resolution mode (see the **DCRES** command).

Array names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore character ("_"), or the question mark ("?"). Array names must not be the same as HP 3245A commands or parameters or stored state names.

Remarks

APPLY DCMEMV Operation

When **APPLY DCMEMV** is executed, the value in the first element of *array_name* is transferred to the channel and the equivalent voltage level is output. Note that a trigger is NOT required to output the first voltage level.

The remaining voltage levels are then output, one at a time, as high-to-low triggers are received from the trigger source selected by **TRIGIN**. After the last voltage level in the array is output, the next trigger causes wrap-around and the first voltage level is again output.

Using APPLY DCMEMV With Autorange (ARANGE)

If autorange is enabled (**ARANGE ON** or **RANGE AUTO**), **APPLY DCMEMV** searches *array_name* elements and selects the lowest voltage range which will provide maximum accuracy for the largest value in the array. If autorange is disabled, the instrument retains the range selected by the **RANGE** command and generates an error if any value in *array_name* attempts to exceed the present range.

Low-Resolution Mode is Used (DCRES)

APPLY DCMEMV automatically generates DC voltages using the low-resolution mode (**DCRES LOW**). If the high-resolution mode (**DCRES HIGH**) is in effect when **APPLY DCMEMV** is selected, the channel temporarily goes to the low-resolution mode. When **APPLY DCMEMV** completes, the channel returns to the high-resolution mode.

Triggering the Channel (TRIGIN)

APPLY DCMEMV outputs DC voltage values in response to triggers received. **TRIGIN** selects the trigger source which activates the **APPLY DCMEMV** command. Refer to the **TRIGIN** command for details.

Related Commands

APPLY DCMEMI, DCRES, OUTPUT?, TERM, TRIGIN, USE

Example

Example: Triggered-Sequence DC Voltage Outputs (APP_DCMV)

This program outputs a triggered DC voltage sequence from channel A consisting of four voltage values: 1.1, 2.2, 3.3, and 4.4 VDC. When the program executes, **APPLY DCMEMV** outputs the first value in the array (1.1 VDC). The remaining voltages are output, one at a time, at 3 second intervals when **TRIGIN SGL** (a software trigger) is executed. Note that the first output level (1.1 VDC) is output without a trigger.

```
10    !file APP_DCMV
20    !
30    CLEAR 709                             !Clear HP 3245A
40    OUTPUT 709;"RST"                      !Reset HP 3245A
50    OUTPUT 709;"SCRATCH"                  !Clear HP 3245A memory
60    !
70    OUTPUT 709;"DIM VOUT(3)"              !Dimension array
80    OUTPUT 709;"FILL VOUT 1.1,2.2,3.3,4.4"  !Enter array values
90    !
100   OUTPUT 709;"SUB V_OUTPUT"             !Begin subroutine
110   OUTPUT 709;"  USE 0"                  !Use channel A
120   OUTPUT 709;"    APPLY DCMEMV 4,VOUT"    !Output triggered DCV
130   OUTPUT 709;"  WAIT 3"                 !Wait 3 seconds
140   OUTPUT 709;"  FOR I = 1 TO 3"         !Begin count loop
150   OUTPUT 709;"    TRIGIN SGL"           !Single trigger
160   OUTPUT 709;"    WAIT 3"               !Wait 3 seconds
170   OUTPUT 709;"  NEXT I"                 !Increment loop
180   OUTPUT 709;"SUBEND"                   !End subroutine
190   !
200   OUTPUT 709;"RUN V_OUTPUT"             !Run subroutine
210   END
```

Description

Apply DC Voltage Output. **APPLY DCV** outputs a DC voltage from the **USE** channel.

Syntax

APPLY DCV *volts*

Parameters

volts Output Voltage. Valid range is -10.25 VDC to +10.25 VDC in the 0Ω output impedance mode or -5.125 VDC to +5.125 VDC in the 50Ω output impedance mode (when terminated with a 50Ω load).

Remarks

DC Voltage Ranges (APPLY DCV)

Voltages can be generated on one of two ranges in the high-resolution mode or on one of seven ranges in the low-resolution mode. Refer to **DCRES** command for details on the resolution modes. The following tables show DC voltage ranges and maximum programmed output values available.

- DC Voltage Ranges (High-Resolution Mode):

Output Impedance	Range	Maximum Programmed Output	Resolution
0Ω	1V	1.25V	1 µV
0Ω	10V	10.25V	10 µV
50Ω	0.5V	0.625V	0.5 µV
50Ω	5V	5.125V	5 µV

- DC Voltage Ranges (Low-Resolution Mode):

Output Impedance	Range	Maximum Programmed Output	Resolution
0Ω	0.15625V	0.15625V	78 µV
0Ω	0.3125V	0.3125V	156 µV
0Ω	0.625V	0.625V	312 µV
0Ω	1.25V	1.25V	625 mV
0Ω	2.5V	2.5V	1.25 mV
0Ω	5V	5V	2.5 mV
0Ω	10V	10V	5 mV
50Ω	0.078125V	0.078125V	39 µV
50Ω	0.15625V	0.15625V	78 µV
50Ω	0.3125V	0.3125V	156 µV
50Ω	0.625V	0.625V	312 µV
50Ω	1.25V	1.25V	625 mV
50Ω	2.5V	2.5V	1.25 mV
50Ω	5V	5V	2.5 mV

Setting the Output Impedance (IMP)

For DC voltage outputs (and AC voltage outputs), you can set the output impedance to 0Ω or 50Ω. In the power-on 0Ω mode, the voltage output from the channel equals the programmed output voltage with or without termination. However, in the 50Ω mode, the actual output voltage equals the programmed output voltage ONLY when the module is terminated with a 50Ω load. Other impedances will cause the output voltage to vary. Refer to the **IMP** command for details on setting the output impedance.

The Autorange Function (ARANGE)

When the autorange function is enabled (**ARANGE ON**), the HP 3245A selects the lowest voltage or current range which will provide maximum accuracy for the desired output level. An error is generated if you attempt to set the output level greater than 100% of the highest voltage or current range available.

When the autorange function is disabled (**ARANGE OFF**), the HP 3245A retains the present range regardless of the programmed output value. If the programmed value attempts to exceed 100% of the present range, an error is generated.

Related Commands

ARANGE, DCRES, IMP, OUTPUT?, TERM, USE

Examples

Example: DC Voltage Output (APPL_DCV)

This program outputs 3.5 VDC from channel A.

```
10   !file APPL_DCV
20   !
30   CLEAR 709                     !Clear HP 3245A
40   OUTPUT 709;"RST"              !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"          !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"            !Use channel A
80   OUTPUT 709;"APPLY DCV 3.5"    !Output DC voltage @3.5 VDC
90   END
```

Description

Apply Current Ramp Waveform. APPLY RPI outputs an AC ramp waveform from the **USE** channel.

Syntax

APPLY RPI *pp_amplitude*

Parameters

pp_amplitude

Peak-to-peak amplitude between 0.00001 A and 0.2 Amps.

Remarks

AC Current Ranges (APPLY RPI)

AC ramp current waveforms can be generated on one of four ranges in the 0Ω or 50Ω output impedance mode. The following table shows the AC current ranges and maximum programmed output values available in each output impedance mode. You can set the output amplitude from 5% to 100% of the specified range.

Range	Maximum Programmed Output	Resolution
0.0002 A	0.0002 A	50 nA
0.002 A	0.002 A	500 nA
0.02 A	0.02 A	5 µA
0.2 A	0.2 A	50 µA

The Autorange Function (ARANGE)

When the autorange function is enabled (**ARANGE ON**), the HP 3245A automatically selects the lowest voltage or current range which will provide maximum accuracy for the desired output level. An error is generated if you attempt to set the output level greater than 100% of the highest voltage or current range available. For AC waveforms, an error is also generated if you attempt to set the peak-to-peak amplitude less than 5% of the lowest peak-to-peak range.

When the autorange function is disabled (**ARANGE OFF**), the instrument retains the present range regardless of the programmed output value. If the programmed value attempts to exceed 100% of the present range, an error is generated. For AC waveforms, an error is also generated if you attempt to set the peak-to-peak amplitude less than 5% of the present peak-to-peak range.

Selecting the Output Frequency (FREQ)

At power-on, the output frequency for AC ramp waveforms is 1000 Hz. The output frequency can be varied from 0 to 1 MHz with 0.001 Hz resolution. However, ramp waveform performance is not specified above 100 kHz and will degrade substantially above this frequency. Refer to the **FREQ** command for details on setting the output frequency.

Setting the DC Offset Voltage (DCOFF)

At power-on, the DC offset voltage for AC waveforms is 0 volts. The offset voltage can be changed using the **DCOFF** command. The offset can be a positive or negative number such that the peak AC value of the output waveform, plus the magnitude of the offset, does not exceed 50% of the selected peak-to-peak range. Refer to the **DCOFF** command for details on selecting the DC offset voltage.

Setting the Duty Cycle (DUTY)

At power-on, the duty cycle for ramp waveforms is set to 50%. You can vary the duty cycle from 5% to 95% for frequencies up to 100 kHz. Refer to the **DUTY** command for details on setting the duty cycle.

PP to RMS Conversion

APPLY RPI generates an AC ramp current waveform with peak-to-peak (PP) values set by *pp_amplitude*. For ramp current outputs measured with an RMS multimeter, use $I_{PP} = 3.464 * I_{RMS}$ to convert an RMS reading to an equivalent PP value.

Related Commands

ARANGE, DCOFF, DUTY, FREQ, IMP, OUTPUT?, TERM, USE

Example

Example: AC Ramp Waveform Current Output (APPL_RPI)

This program outputs a 0.005 A PP (5 mA) ramp waveform from channel A.

```
10   !file APPL_RPI
20   !
30   CLEAR 709                        !Clear HP 3245A
40   OUTPUT 709;"RST"                 !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"             !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"               !Use channel A
80   OUTPUT 709;"APPLY RPI 0.005"     !Output 0.005 A PP ramp waveform
90   END
```

Description

Apply Voltage Ramp Waveform. APPLY RPV outputs an AC ramp voltage waveform from the **USE** channel.

Syntax

APPLY RPV *pp_amplitude*

Parameters

pp_amplitude

Peak-to-peak amplitude between 0.03125 Vac (PP) and 20 Vac (PP) in the 0Ω output impedance mode or between 0.015625 Vac (PP) and 10 Vac (PP) in the 50Ω mode (when terminated with a 50Ω load).

Remarks

AC Voltage Ranges (APPLY RPV)

AC ramp voltage waveforms can be generated on one of seven ranges in the 0Ω or 50Ω output impedance mode. The following table shows the AC voltage ranges and maximum programmed output values available in each output impedance mode. You can set the output amplitude from 10% to 100% of the specified range.

Output Impedance	Range (Volts P-P)	Maximum Programmed Output (Volts P-P)	Resolution
0Ω	0.3125V	0.3125V	156 μV
0Ω	0.625V	0.625V	312 μV
0Ω	1.25V	1.25V	625 μV
0Ω	2.5V	2.5V	1.25 mV
0Ω	5V	5V	2.5 mV
0Ω	10V	10V	5 mV
0Ω	20V	20V	10 mV
50Ω	0.15625V	0.15625V	78 μV
50Ω	0.3125V	0.3125V	156 μV
50Ω	0.625V	0.625V	312 μV
50Ω	1.25V	1.25V	625 μV
50Ω	2.5V	2.5V	1.25 mV
50Ω	5V	5V	2.5 mV
50Ω	10V	10V	5 mV

Setting the Output Impedance (IMP)

The output impedance for either channel can be set to 0Ω or 50Ω. In the 0Ω (power-on) mode, the voltage output is the same as the programmed output voltage with or without termination. However, in the 50Ω mode, the actual output voltage is equal to the programmed output voltage ONLY when the output is terminated with a 50Ω load. Other impedances will cause the output voltage to vary. Refer to the **IMP** command for details on setting the output impedance.

The Autorange Function (ARANGE)

When the autorange function is enabled (**ARANGE ON**), the HP 3245A automatically selects the lowest voltage or current range which will provide maximum accuracy for the desired output level. An error is generated if you attempt to set the output level greater than 100% of the highest voltage or current range available. For AC waveforms, an error is also generated if you attempt to set the peak-to-peak amplitude less than 10% of the lowest peak-to-peak range.

When the autorange function is disabled (**ARANGE OFF**), the instrument retains the present range regardless of the programmed output value. If the programmed value attempts to exceed 100% of the present range, an error is generated. For AC waveforms, an error is also generated if you attempt to set the peak-to-peak amplitude less than 10% of the present peak-to-peak range.

Selecting the Output Frequency (FREQ)

At power-on, the output frequency for AC waveforms is 1000 Hz. The output frequency can be varied from 0 to 1 MHz with 0.001 Hz resolution. However, ramp waveform performance is not specified above 100 kHz and will degrade substantially above this frequency. Refer to the **FREQ** command for details on setting the output frequency.

Setting the DC Offset Voltage (DCOFF)

At power-on, the DC offset voltage for AC waveforms is 0 volts. The offset voltage can be changed using the **DCOFF** command. The offset can be a positive or negative number such that the peak AC value of the output waveform, plus the magnitude of the offset, does not exceed 50% of the selected peak-to-peak range. Refer to the **DCOFF** command for details on selecting the DC offset voltage.

Setting the Duty Cycle

At power-on, the duty cycle for ramp waveforms is set to 50%. You can vary the duty cycle from 5% to 95% for frequencies up to 100 kHz. Refer to the **DUTY** command for details on setting the duty cycle.

PP to RMS Conversion

APPLY RPV generates a voltage ramp waveform with peak-to-peak (PP) values set by *pp_amplitude*. For ramp voltage outputs measured with an RMS multimeter, use $V_{PP} = 3.464*V_{RMS}$ to convert an RMS reading to an equivalent PP value.

Related Commands

ARANGE, DCOFF, DUTY, FREQ, IMP, OUTPUT?, TERM, USE

Example

Example: AC Ramp Waveform Voltage Output (APPL_RPV)

This program outputs a 4.5 Vac PP ramp waveform from channel A.

```
10   !file APPL_RPV
20   !
30   CLEAR 709                     !Clear HP 3245A
40   OUTPUT 790;"RST"              !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"          !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"            !Use channel A
80   OUTPUT 709;"APPLY RPV 4.5"    !Output 4.5 Vac PP ramp waveform
90   END
```

APPLY SQI

Description

Apply Current Square Waveform. APPLY SQI outputs an AC current square waveform from the **USE** channel.

Syntax

APPLY SQI *pp_amplitude*

Parameters

pp_amplitude Peak-to-peak amplitude between 0.00001 A and 0.2 Amps.

Remarks

AC Current Ranges (APPLY SQI)

AC current square waveforms can be generated on one of four ranges in the 0Ω or 50Ω output impedance mode. The following table shows the AC current ranges and maximum programmed output values available in each output impedance mode. You can set the output amplitude from 5% to 100% of the specified range.

Range	Maximum Programmed Output	Resolution
0.0002 A	0.0002 A	50 nA
0.002 A	0.002 A	500 nA
0.02 A	0.02 A	5 μA
0.2 A	0.2 A	50 μA

The Autorange Function (ARANGE)

When the autorange function is enabled (**ARANGE ON**), the HP 3245A automatically selects the lowest voltage or current range which will provide maximum accuracy for the desired output level. An error is generated if you attempt to set the output level greater than 100% of the highest voltage or current range available. For AC waveforms, an error is also generated if you attempt to set the peak-to-peak amplitude less than 5% of the lowest peak-to-peak range.

When the autorange function is disabled (**ARANGE OFF**), the instrument retains the present range regardless of the programmed output value. If the programmed value attempts to exceed 100% of the present range, an error is generated. For AC waveforms, an error is also generated if you attempt to set the peak-to-peak amplitude less than 5% of the present peak-to-peak range.

Selecting the Output Frequency (FREQ)

At power-on, the output frequency for AC waveforms is 1000 Hz. The output frequency can be varied from 0 to 1 MHz with 0.001 Hz resolution. Refer to the **FREQ** command for details on setting the output frequency.

Setting the DC Offset Voltage (DCOFF)

At power-on, the DC offset voltage for AC waveforms is 0 volts. The offset voltage can be changed using the **DCOFF** command. The offset can be a positive or negative number such that the peak AC value of the output waveform, plus the magnitude of the offset, does not exceed 50% of the selected peak-to-peak range. Refer to the **DCOFF** command for details on selecting the DC offset voltage.

Setting the Duty Cycle (DUTY)

At power-on, the duty cycle for square waveforms is set to 50%. You can vary the duty cycle from 5% to 95% for frequencies up to 100 kHz. Refer to the **DUTY** command for details on setting the duty cycle.

PP to RMS Conversion

APPLY SQI generates an AC current square waveform with peak-to-peak (PP) values set by *pp_amplitude*. For outputs measured with an RMS multimeter, use IPP = 2*IRMS.

Related Commands

ARANGE, DCOFF, DUTY, FREQ, IMP, OUTPUT?, TERM, USE

Example

Example: AC Square Waveform Current Output (APPL_SQI)

This program outputs a 0.005 A PP (5 mA) square waveform from channel A.

```
10   !file APPL_SQI
20   !
30   CLEAR 709                    !Clear HP 3245A
40   OUTPUT 709;"RST"             !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"         !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"           !Use channel A
80   OUTPUT 709;"APPLY SQI 0.005" !Output 0.005 A PP square waveform
90  END
```

APPLY SQV

Description

Apply Voltage Square Waveform. APPLY SQV outputs an AC voltage square waveform from the **USE** channel.

Syntax

APPLY SQV *pp_amplitude*

Parameters

pp_amplitude

Peak-to-peak amplitude between 0.03125 Vac (PP) and 20 Vac (PP) in the 0Ω output impedance mode or between 0.015625 Vac (PP) and 10 Vac (PP) in the 50Ω mode (when terminated with a 50Ω load).

Remarks

AC Voltage Ranges (APPLY SQV)

AC voltage square waveforms can be generated on one of seven ranges in the 0Ω or 50Ω output impedance mode. The following table shows the AC voltage ranges and maximum programmed output values available in each output impedance mode. You can set the output amplitude from 10% to 100% of the specified range.

Output Impedance	Range (Volts P-P)	Maximum Programmed Output (Volts P-P)	Resolution
0Ω	0.3125V	0.3125V	156 µV
0Ω	0.625V	0.625V	312 µV
0Ω	1.25V	1.25V	625 µV
0Ω	2.5V	2.5V	1.25 mV
0Ω	5V	5V	2.5 mV
0Ω	10V	10V	5 mV
0Ω	20V	20V	10 mV
50Ω	0.15625V	0.15625V	78 µV
50Ω	0.3125V	0.3125V	156 µV
50Ω	0.625V	0.625V	312 µV
50Ω	1.25V	1.25V	625 µV
50Ω	2.5V	2.5V	1.25 mV
50Ω	5V	5V	2.5 mV
50Ω	10V	10V	5 mV

Setting the Output Impedance (IMP)

The output impedance for either channel can be set to 0Ω or 50Ω. In the 0Ω (power-on) mode, the voltage output is the same as the programmed output voltage with or without termination. However, in the 50Ω mode, the actual output voltage is equal to the programmed output voltage ONLY when the output is terminated with a 50Ω load. Other impedances will cause the output voltage to vary. Refer to the **IMP** command for details on setting the output impedance.

The Autorange Function (ARANGE)

When the autorange function is enabled (**ARANGE ON**), the HP 3245A automatically selects the lowest voltage or current range which will provide maximum accuracy for the desired output level. An error is generated if you attempt to set the output level greater than 100% of the highest voltage or current range available. For AC waveforms, an error is also generated if you attempt to set the peak-to-peak amplitude less than 10% of the lowest peak-to-peak range.

When the autorange function is disabled (**ARANGE OFF**), the instrument retains the present range regardless of the programmed output value. If the programmed value attempts to exceed 100% of the present range, an error is generated. For AC waveforms, an error is also generated if you attempt to set the peak-to-peak amplitude less than 10% of the present peak-to-peak range.

Selecting the Output Frequency (FREQ)

At power-on, the output frequency for AC waveforms is 1000 Hz. The output frequency can be varied from 0 to 1 MHz with 0.001 Hz resolution. Refer to the **FREQ** command for details on setting the output frequency.

Setting the DC Offset Voltage (DCOFF)

At power-on, the DC offset voltage for AC waveforms is 0 volts. The offset voltage can be changed using the **DCOFF** command. The offset can be a positive or negative number such that the peak AC value of the output waveform, plus the magnitude of the offset, does not exceed 50% of the selected peak-to-peak range. Refer to the **DCOFF** command for details on selecting the DC offset voltage.

Setting the Duty Cycle (DUTY)

At power-on, the duty cycle for square waveforms is set to 50%. You can vary the duty cycle from 5% to 95% for frequencies up to 100 kHz. Refer to the **DUTY** command for details on setting the duty cycle.

PP to RMS Conversion

APPLY SQV generates an AC voltage square waveform with peak-to-peak (PP) value set by *pp_amplitude*. For outputs measured with an RMS multimeter, use $V_{PP} = 2*V_{RMS}$.

Related Commands

ARANGE, DCOFF, DUTY, FREQ, IMP, OUTPUT?, TERM, USE

Example

Example: AC Square Waveform Voltage Output (APPL_SQV)

This program outputs a 4.5 Vac PP square waveform from channel A.

```
10   !file APPL_SQV
20   !
30   CLEAR 709                   !Clear HP 3245A
40   OUTPUT 709;"RST"            !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"        !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"          !Use channel A
80   OUTPUT 709;"APPLY SQV 4.5"  !Output 4.5 Vac PP square waveform
90   END
```

Description

Apply Current Arbitrary Waveform. **APPLY WFI** outputs an arbitrary current waveform from the **USE** channel.

Syntax

APPLY WFI *pp_amplitude* [*array_name*]

Parameters

pp_amplitude

Peak-to-peak amplitude from 0.00001 A to 0.2 A.

array_name

Specifies the name of the REAL or INTEGER array containing 2048 points which define the waveform. Each REAL array element must have a value between -1 and +1, inclusive. If *array_name* is defined as an INTEGER array, each element must be converted to a hardware integer format before executing **APPLY WFI**. Refer to the **FILLAC**, **FILLRP**, and **FILLWF** commands for details.

Array names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore characters ("_"), or the question mark ("?"). Array names must not be the same as HP 3245A commands or parameters or stored state names. The default *array_name* is the name of the array most recently specified with **APPLY WFI**. (When **APPLY WFI** is used with only the *pp_amplitude* parameter, a new array is not downloaded - only the amplitude is changed.)

Remarks

AC Current Ranges (APPLY WFI)

AC arbitrary current waveforms can be generated on one of four ranges in the 0Ω or 50Ω output impedance mode. The following table shows the AC current ranges and maximum programmed output values available in each output impedance mode. You can set the output amplitude from 5% to 100% of the specified range.

Range	Maximum Programmed Output	Resolution
0.0002 A	0.0002 A	50 nA
0.002 A	0.002 A	500 nA
0.02 A	0.02 A	5 µA
0.2 A	0.2 A	50 µA

The Autorange Function (ARANGE)

When the autorange function is enabled (**ARANGE ON**), the HP 3245A automatically selects the lowest current range which will provide maximum accuracy for the desired output level. An error is generated if you attempt to set the output level greater than 100% of the highest current range available. For AC waveforms, an error is also generated if you attempt to set the peak-to-peak amplitude less than 10% of the lowest peak-to-peak range.

When the autorange function is disabled (**ARANGE OFF**), the instrument retains the present range regardless of the programmed output value. If the programmed value attempts to exceed 100% of the present range, an error is generated. For AC waveforms, an error is also generated if you attempt to set the peak-to-peak amplitude less than 10% of the present peak-to-peak range.

Selecting the Output Frequency (FREQ)

At power-on, the output frequency for AC waveforms is 1000 Hz. The output frequency can be varied from 0 to 1 MHz with 0.001 Hz resolution. Refer to the **FREQ** command for details on setting the output frequency.

Setting the DC Offset Voltage (DCOFF)

At power-on, the DC offset voltage for AC waveforms is 0 volts. The offset voltage can be changed using the **DCOFF** command. The offset can be a positive or negative number such that the peak AC value of the output waveform, plus the magnitude of the offset, does not exceed 50% of the selected peak-to-peak range. Refer to the **DCOFF** command for details on selecting the DC offset voltage.

Related Commands

DCOFF, FILLAC, FILLRP, FILLWF, FREQ, IMP, OUTPUT?, TERM, USE

Example

Example: Generating a Cosine Waveform (APPL_WFI)

This program defines and applies a 2 kHz cosine waveform with an amplitude of 0.005 A (5 mA) PP from channel A.

```
10   !file APPL_WFI
20   !
30   CLEAR 709                                        !Clear HP 3245A
40   OUTPUT 709;"RST"                                 !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"                             !Clear HP 3245A memory
60   !
70   OUTPUT 709;"REAL CARRAY(2047),PI"                !Dim REAL array, var
80   OUTPUT 709;"PI = 3.1415925"                      !Assign value to PI
90   !
100  OUTPUT 709;"SUB DEF_ARRAY"                       !Begin subroutine
110  OUTPUT 709;"  FOR I = 0 TO 2047"                 !Begin count loop
120  OUTPUT 709;"    CARRAY(I) = COS(2*PI*I/2048)"    !Compute array values
130  OUTPUT 709;"  NEXT I"                            !Increment loop
140  OUTPUT 709;"SUBEND"                              !End subroutine
150  OUTPUT 709;"CALL DEF_ARRAY"                      !Call subroutine
160  !
170  OUTPUT 709;"USE 0"                               !Use channel A
180  OUTPUT 709;"FREQ 2E3"                            !Change freq to 2 kHz
190  OUTPUT 709;"APPLY WFI 0.005,CARRAY"              !Output cosine waveform
200  END
```

Description

Apply Voltage Arbitrary Waveform. **APPLY WFV** outputs an arbitrary voltage waveform from the **USE** channel.

Syntax

APPLY WFV *pp_amplitude* [, *array_name*]

Parameters

pp_amplitude

Peak-to-peak amplitude from 0.03125 Vac (PP) to 20 Vac (PP) in the 0Ω output impedance mode or from 0.015625 Vac (PP) to 10 Vac (PP) in the 50Ω mode (when terminated with a 50Ω load).

array_name

Name of a REAL or INTEGER array containing 2048 points which define the waveform. Each REAL array element must have a value between -1 and +1, inclusive. If *array_name* is defined as an INTEGER array, each element must be converted to a hardware integer format (with **FILLAC**, **FILLRP**, or **FILLWF**) before executing **APPLY WFV**.

Array names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore characters ("_"), or the question mark ("?"). Array names must not be the same as HP 3245A commands or parameters or stored state names. The default *array_name* is the name of the array last specified with **APPLY WFV**. (When **APPLY WFV** is used with only the *pp_amplitude* parameter, a new array is not downloaded - only the amplitude is changed.)

Remarks

AC Voltage Ranges (APPLY WFV)

Arbitrary waveforms can be generated using one of seven ranges in the 0Ω or 50Ω output impedance mode, as shown in the following table. Output amplitude can be set from 10% to 100% of the specified range.

Output Impedance	Range (Volts P-P)	Maximum Programmed Output (Volts P-P)	Resolution
0Ω	0.3125V	0.3125V	156 µV
0Ω	0.625V	0.625V	312 µV
0Ω	1.25V	1.25V	625 µV
0Ω	2.5V	2.5V	1.25 mV
0Ω	5V	5V	2.5 mV
0Ω	10V	10V	5 mV
0Ω	20V	20V	10 mV
50Ω	0.15625V	0.15625V	78 µV
50Ω	0.3125V	0.3125V	156 µV
50Ω	0.625V	0.625V	312 µV
50Ω	1.25V	1.25V	625 µV
50Ω	2.5V	2.5V	1.25 mV
50Ω	5V	5V	2.5 mV
50Ω	10V	10V	5 mV

Setting the Output Impedance (IMP)

The output impedance for either channel can be set to 0Ω or 50Ω. In the 0Ω (power-on) mode, the voltage output is the same as the programmed output voltage with or without termination. However, in the 50Ω mode, the actual output voltage is equal to the programmed output voltage ONLY when the output is terminated with a 50Ω load. Refer to the **IMP** command for details on setting the output impedance.

The Autorange Function (ARANGE)

When the autorange function is enabled (**ARANGE ON**), the HP 3245A automatically selects the lowest voltage or current range which will provide maximum accuracy for the desired output level. An error is generated if you attempt to set the output level greater than 100% of the highest voltage or current range available. For AC waveforms, an error is also generated if you attempt to set the peak-to-peak amplitude less than 10% of the lowest peak-to-peak range.

When the autorange function is disabled (**ARANGE OFF**), the instrument retains the present range regardless of the programmed output value. If the programmed value attempts to exceed 100% of the present range, an error is generated. For AC waveforms, an error is also generated if you attempt to set the peak-to-peak amplitude less than 10% of the present peak-to-peak range.

Selecting the Output Frequency (FREQ)

At power-on, the output frequency for AC waveforms is 1000 Hz. The output frequency can be varied from 0 to 1 MHz with 0.001 Hz resolution. Refer to the **FREQ** command for details on setting the output frequency.

Setting the DC Offset Voltage (DCOFF)

At power-on, the DC offset voltage for AC waveforms is 0 volts. The offset voltage can be changed using the **DCOFF** command. The offset can be a positive or negative number such that the peak AC value of the output waveform, plus the magnitude of the offset, does not exceed 50% of the selected peak-to-peak range. Refer to the **DCOFF** command for details on selecting the DC offset voltage.

Related Commands

DCOFF, FILLAC, FILLRP, FILLWF, FREQ, IMP, OUTPUT?, TERM, USE

Example

Example: Cosine Waveform Generation (APPL_WFV)

This program defines and applies a 2 kHz cosine waveform with an amplitude of 2.4 Vac PP from channel A. Since precomputing (with a **FILLWF** command) is not done, approximately 625 msec scaling time + 50 msec download time is required to output the waveform. However, **FILLWF** can be used to reduce the time required to output the waveform. See the **FILLWF** command for details.

```
10   !file APPL_WFV
20   !
30   CLEAR 709                                       !Clear HP 3245A
40   OUTPUT 709;"RST"                                !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"                            !Clear HP 3245A memory
60   !
70   OUTPUT 709;"REAL CARRAY(2047),PI"               !Dim REAL array, var
80   OUTPUT 709;"PI = 3.1415925"                     !Assign value to PI
90   !
100  OUTPUT 709;"SUB DEF_ARRAY"                      !Begin subroutine
110  OUTPUT 709;"  FOR I = 0 TO 2047"                !Begin count loop
120  OUTPUT 709;"    CARRAY(I) = COS(2*PI*I/2048)"   !Compute array values
130  OUTPUT 709;"  NEXT I"                           !Increment loop
140  OUTPUT 709;"SUBEND"                             !End subroutine
150  OUTPUT 709;"CALL DEF_ARRAY"                     !Call subroutine
160  !
170  OUTPUT 709;"USE 0"                              !Use channel A
180  OUTPUT 709;"FREQ 2E3"                           !Change freq to 2 kHz
190  OUTPUT 709;"APPLY WFV 2.4,CARRAY"               !Output cosine waveform
200  END
```

ARANGE/ARANGE?

Description

Set/Read Autorange. **ARANGE** enables or disables the autorange function on the **USE** channel. **ARANGE?** returns the autorange setting (OFF/ON) for the **USE** channel.

Syntax

ARANGE [*mode*]

ARANGE?

Parameters

mode

The *mode* parameters follow. Power-on, reset, and default *mode* = ON.

mode	Definition
OFF	Disables autorange function.
ON	Enables autorange function.

Remarks

Using ARANGE

When the autorange function is enabled (**ARANGE ON**), the lowest voltage or current range is selected which will provide maximum accuracy for the desired output level. An error is generated if you attempt to set the output level greater than 100% of the highest voltage or current range available. For AC waveforms, an error is also generated if you attempt to set the peak-to-peak amplitude less than 10% of the lowest peak-to-peak range.

When the autorange function is disabled (**ARANGE OFF**), the HP 3245A retains the present range regardless of the programmed output value. If the programmed value attempts to exceed 100% of the present range, an error is generated. For AC waveforms, an error is also generated if you attempt to set the peak-to-peak amplitude less than 10% of the present peak-to-peak voltage range (less than 5% of the present peak-to-peak current range).

Autorange Function Automatically Enabled

At power-on, the autorange function is enabled (**ARANGE ON**). Autoranging is automatically enabled when commands which change the DC resolution mode(**DCRES**), output impedance (**IMP**), or output function (**APPLY**) commands are executed. Also, if **ARANGE OFF** is set immediately after power-on, the 1V range is set.

Query Command (ARANGE?)

The **ARANGE?** command returns the autorange setting (OFF/ON) of the **USE** channel (channel A or B).

Related Commands

APPLYs, DCRES, FUNC, IMP, RANGE, TERM, USE

Example

Example: Using Autorange Function (ARANGE)

This program first sets DCV output (line 70) and turns autorange OFF (line 80). Then, changing the output function to ACV (line 130) automatically turns autorange ON, since the output function is changed from DCV to ACV.

```
10   !file ARANGE
20   !
30   CLEAR 709                        !Clear HP 3245A
40   OUTPUT 709;"RST 0"               !Reset channel A
50   !
60   OUTPUT 709;"USE 0"               !Use channel A
70   OUTPUT 709;"  APPLY DCV 10"      !Output 10 VDC
80   OUTPUT 709;"  ARANGE OFF"        !Disable autorange
90   OUTPUT 709;"  ARANGE?"
100  ENTER 709;A$
110  PRINT "Autorange Mode = ";A$
120  OUTPUT 709;"WAIT 1"              !Wait one sec
130  OUTPUT 709;"  APPLY ACV .05"     !Output 1 VDC
140  OUTPUT 709;"  ARANGE?"           !Query autorange mode
150  ENTER 709;B$                     !Enter mode
160  PRINT "Autorange Mode = ";B$     !Display mode
170  END
```

A typical return follows. Note that autorange was disabled (fixed range set) for the DCV output, but autorange was again enabled for the ACV output function, since the function changed.

```
Autorange Mode = OFF
Autorange Mode = ON
```

ATN

Description

Arctangent. Returns the value of the angle (in radians) which has a tangent equal to the argument.

Syntax

ATN *(argument)*

Parameters

argument — The *argument* parameter must be a number or numeric expression (enclosed in parentheses).

Remarks

Related Commands

COS, SIN

Example

Example: Using the ATN Function (ATN)

This program computes the arctangent of 0.2018 and displays the result (.19912573 radians) on the controller CRT.

```
10   !file ATN
20   !
30   OUTPUT 709;"VREAD ATN(0.2018)"     !Compute the artangent
40   ENTER 709;A                        !Enter result
50   PRINT "Arctangent = ";A;"radians"  !Display result
60   END
```

A typical return is:

```
   Arctangent = .19912573 radians
```

Description

Beep. Enables or disables the HP 3245A beeper mode. When the beeper mode is enabled, a beep occurs when an error is generated. When disabled, error messages are displayed on the front panel but the beep is suppressed.

Syntax

BEEP [*mode*]

Parameters

mode The *mode* parameters follow. Default *mode* = ONCE.

mode Parameter	Definition
OFF	Disables beeper mode.
ON	Enables beeper mode.
ONCE	Beep once then return to previous mode (OFF or ON).

Remarks

BEEP ONCE Overrides Present Mode

The **BEEP ONCE** command overrides the present ON/OFF mode of the **BEEP** command and produces a single beep. The beep occurs even if the beeper mode is disabled (**BEEP OFF**). After a single beep, the beeper mode returns to the previous mode (ON or OFF).

Beeper Mode is Stored in Continuous Memory

The beeper mode is stored in continuous memory and is retained when power is removed from the HP 3245A.

BEEP Does not Affect Power-On Sequence

The HP 3245A signals the end of its power-on test with a tone. The **BEEP** command does not affect this tone.

Example

Example: Produce a Single Beep

This program temporarily disables the beeper mode, produces one beep, and then returns to the **BEEP OFF** mode.

```
10  OUTPUT 709;"BEEP OFF"     !Disable beeper mode
20  OUTPUT 709;"BEEP ONCE"    !Beep once and return to BEEP OFF mode
30  END
```

BINAND

Description

Binary AND. Returns the value of the bit-by-bit logical AND of the specified arguments.

Syntax

BINAND (*argument*, *argument*)

Parameters

argument

Both *argument* parameters must be integer numbers or numeric expressions (enclosed in parentheses) in the range -32768 to +32767.

Remarks

BINAND Operation

Within the HP 3245A processor, each bit in one *argument* is logically ANDed with the corresponding bit in the other *argument*, as shown in the following truth table.

A B	A AND B
0 0	0
0 1	0
1 0	0
1 1	1

Related Commands

AND, BINCMP, BINEOR, BINIOR

Example

Example: Using the BINAND Function

This program computes the **BINAND** value of 12 and 9 and displays the result (8) on the controller CRT. The **BINAND** arithmetic is:

```
12 = 0000 0000 0000 1100
 9 = 0000 0000 0000 1001
_________________________
 8 = 0000 0000 0000 1000
```

```
10  OUTPUT 709;"VREAD BINAND(12,9)"   !Compute and read BINAND value
20  ENTER 709;A                       !Enter result
30  PRINT "BINAND Value = ";A         !Display result
40  END
```

A typical return is:

```
BINAND Value = 8
```

Description

Binary Complement. Returns the binary complement of the specified argument.

Syntax

BINCMP (*argument*)

Parameters

argument

The *argument* parameter must be an integer number or numeric integer (enclosed in parentheses) in the range -32768 to +32767.

Remarks

BINCMP Operation

Within the HP 3245A processor, each bit in *argument* is complemented and the resulting value is returned.

Related Commands

BINAND, BINEOR, BINIOR

Example

Example: Using the BINCMP Function

The following program computes the **BINCMP** value of -13 and displays the result (12) on the controller CRT. The **BINCMP** arithmetic is:

```
-13 = 1111 1111 1111 0011
 12 = 0000 0000 0000 1100 = Complement
```

```
10  OUTPUT 709;"VREAD BINCMP(-13)"   !Compute and read BINCMP value
20  ENTER 709;A                      !Enter result
30  PRINT "BINCMP Value = ";A        !Print result
40  END
```

A typical return is:

```
BINCMP Value = 12
```

BINEOR

Description

Binary Exclusive-OR. Returns the value of the bit-by-bit logical exclusive-OR of the specified arguments.

Syntax

BINEOR (*argument, argument*)

Parameters

argument

Both *argument* parameters must be integer numbers or numeric expressions (enclosed in parentheses) in the range -32768 to +32767.

Remarks

BINEOR Operation

Within the HP 3245A processor, each bit in one *argument* is exclusive-ORed with the corresponding bit in the second *argument* as shown in the following truth table:

A B	A XOR B
0 0	0
0 1	1
1 0	1
1 1	0

Related Commands

BINAND, BINCMP, BINIOR

Example

Example: Using the BINEOR Function

The following program computes the **BINEOR** value of 12 and -9 and displays the result (-5) on the controller CRT. The **BINEOR** arithmetic is:

```
12 = 0000 0000 0000 1100
-9 = 1111 1111 1111 0111
------------------------
-5 = 1111 1111 1111 1011
```

```
10  OUTPUT 709;"VREAD BINEOR(12,-9)"   !Compute and read BINEOR value
20  ENTER 709;A                        !Enter result
30  PRINT "BINEOR Value = ";A          !Print result
40  END
```

A typical return is:

```
BINEOR Value = -5
```

Description

Binary Inclusive-OR. Returns the value of the bit-by-bit logical inclusive-OR of the specified arguments.

Syntax

BINIOR (*argument*, *argument*)

Parameters

argument

Both *argument* parameters must be integer numbers or numeric expressions (enclosed in parentheses) in the range -32768 to +32767.

Remarks

BINIOR Operation

Within the HP 3245A processor, each bit in one *argument* is inclusive-ORed with the corresponding bit in the other *argument*, as shown in the following truth table:

A B	A OR B
0 0	0
0 1	1
1 0	1
1 1	1

Related Commands

BINAND, BINCMP, BINEOR, OR

Example

Example: Using the BINIOR Function

The following program computes the **BINIOR** value of 12 and 9 and displays the result (13) on the system controller. The **BINIOR** arithmetic is:

```
12 = 0000 0000 0000 1100
 9 = 0000 0000 0000 1001
_________________________
13 = 0000 0000 0000 1101
```

```
10  OUTPUT 709;"VREAD BINIOR(12,9)"     !Compute and read BINIOR value
20  ENTER 709;A                         !Enter result
30  PRINT "BINIOR Value = ";A           !Display result
40  END
```

A typical return is:

```
BINIOR Value = 13
```

BIT

Description

Read Bit Value. Returns "0" or "1" representing the logic value of the specified bit of the argument.

Syntax

BIT (*argument*, *bit_position*)

Parameters

argument

The *argument* parameter is a number or numeric expression (enclosed in parentheses) in the range -32768 to +32767. Within the HP 3245A processor, *argument* is represented as a 16-bit, 2's complement integer.

bit_position

The *bit_position* parameter is a number or numeric expression (enclosed in parentheses) between 0 and 15. Bit 0 represents the least significant bit (LSB) and bit 15 represents the most significant bit (MSB).

Remarks

Data Returned

The **BIT** command returns a "0" or "1" indicating the logic state of the specified bit of the *argument*.

Example

Example: Using the BIT Function

The following example determines the value (0 or 1) of bit 4 in the 16-bit, 2's complement representation of 17. Since 17 = `0000 0000 0001 0001`, the program returns a "1".

```
10  OUTPUT 709;"VREAD BIT(17,4)"   !Read value of bit 4
20  ENTER 709;A                    !Enter result
30  PRINT "Bit Value = ";A         !Display result
40  END
```

A typical return is:

```
    Bit Value = 1
```

BLOCKOUT

Description

Block Output Mode. **BLOCKOUT** applies only to binary output data (as set by **OFORMAT BINARY**). With block output mode enabled, a data header precedes the data output. With block output mode disabled, binary data is output without a preceding header.

Syntax

BLOCKOUT *mode*

Parameters

mode The *mode* parameters are as follows. Power-on/reset *mode* = ON.

mode Parameter	Description
OFF	Binary data is output without preceding header.
ON	IEEE-728 Block A header precedes the binary data.

Remarks

Block Output Disabled (BLOCKOUT OFF)

With **BLOCKOUT OFF**, only the binary number is output (no header or *cr lf*). EOI is sent with the final byte of the binary message if enabled with the **END** command.

Block Output Enabled (BLOCKOUT ON)

At power-on, **BLOCKOUT ON** is set and binary outputs are preceded with an IEEE-728 Block A header and followed by a *cr lf*. The format header consists of four bytes: the # sign, the letter A, and two bytes which indicate the number of bytes of data to follow. EOI is sent with the line feed if enabled with the **END** command.

Related Commands

END, OFORMAT

Example

Example: Disabling Block Output Mode

This program segment sets binary output mode and disables block output mode, so binary data is output without the IEEE-728 Block A header. See the **OFORMAT** command for an example program using **BLOCKOUT** and **OFORMAT**.

```
           .
           .
100  OUTPUT 709;"OFORMAT BINARY"    !Sets binary data output format
110  OUTPUT 709;"BLOCKOUT OFF"      !Output binary data without header
           .
           .
```

CAL

Description

Channel Calibration. Initializes an algorithm to perform a full adjustment of voltage and current ranges.

Refer to the HP 3245A Calibration Manual (P/N 03245-90003) for information on the use of this command.

Description

Read CAL Jumper Setting. Returns the setting of the CAL jumper (ENABLED or DISABLED).

Syntax

CALEN?

Parameters

None.

Remarks

CALEN? Returns CAL Jumper Position

The **CALEN?** command returns the position (ENABLED or DISABLED) of the CAL jumper (J1 or S1) within the HP 3245A. The command returns "1" if the CAL jumper is in the ENABLE position or 0 if the CAL jumper is in the DISABLE position. Calibration with **CAL** can be performed ONLY if the CAL jumper is in the ENABLE position.

Related Commands

CAL

Example

Example: Read CAL Jumper Position

The following program determines the position (ENABLED/DISABLED) of the calibration jumper.

```
10   OUTPUT 709;"CALEN?"    !Read calibration jumper position
20   ENTER 709;A            !Enter result
30   PRINT A                !Display result
40   END
```

If the calibration jumper is in the ENABLE position, "1" is returned. If it is in the DISABLE position, "0" is returned.

CALL

Description

Call Subroutine. Executes the named subroutine and waits for completion before executing other commands (contrast with **RUN**).

Syntax

CALL *sub_name*

Parameters

sub_name

Subroutine name. Subroutine names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore character ("_"), or the question mark ("?"). Subroutine names must not be the same as HP 3245A commands or parameters, previously defined array or variable names, or stored state names.

Remarks

Subroutine Must Be In Memory

The specified subroutine must be stored in HP 3245A memory (using **SUB** and **SUBEND**) before being executed with the **CALL** command.

Executing Nested Subroutines

CALL may be used inside a subroutine to call another subroutine. This provides expanded capability of "nested" subroutines. When using nested subroutines, the calling subroutine is suspended so that only one subroutine is running at a time. Subroutines can be nested up to 10 deep.

Execution Error will Abort Subroutine

If an error occurs during subroutine execution, the subroutine and all nested levels of subroutines are aborted. The cause of the subroutine error is stored in the error register. On error, control is returned to the controller or to the front panel (to where the command was initiated).

Using the USE Command Within Subroutines

If **USE** is executed inside a subroutine called by **CALL**, the HP 3245A retains that assignment after the subroutine completes. For example, if channel A is selected as the **USE** channel within a subroutine called by **CALL**, channel A remains the **USE** channel after the subroutine completes.

CALL vs. RUN Subroutines

When **CALL** is used, the named subroutine runs to completion before executing other commands. When **RUN** is used, the the HP 3245A executes subroutine commands as it finds time between executing commands outside the subroutine.

Related Commands

RUN, SERIAL, SUB, SUBEND, USE

Example

Example: Creating and Calling a Subroutine (CALL)

This program creates and calls subroutine BEEPER. As the program executes, the HP 3245A produces five beeps at 0.5 second intervals.

```
10    !file CALL
20    !
30    CLEAR 709                      !Clear HP 3245A
40    OUTPUT 709;"RST"               !Reset HP 3245A
50    OUTPUT 709;"SCRATCH"           !Clear HP 3245A memory
60    OUTPUT 709;"USE 0"             !Use channel A
70    !
80    OUTPUT 709;"SUB BEEPER"        !Begin subroutine
90    OUTPUT 709;"  FOR I = 1 TO 5"  !Begin count loop
100   OUTPUT 709;"    BEEP ONCE"     !Beep once
110   OUTPUT 709;"    WAIT .5"       !Wait 0.5 seconds
120   OUTPUT 709;"  NEXT I"          !Increment count
130   OUTPUT 709;"SUBEND"            !End subroutine
140   OUTPUT 709;"CALL BEEPER"       !Call subroutine
150   END
```

CALNUM?

Description

Calibration Number Query. Returns the number of times the HP 3245A has been calibrated.

Syntax

CALNUM?

Parameters

None.

Remarks

Data Returned

The **CALNUM?** command returns the number of times the 3245A has been calibrated.

Data Destination

When **CALNUM?** is entered from the front panel, the number of times the instrument has been calibrated is displayed on the front panel display. When **CALNUM?** is executed from the controller, the response is sent to the output buffer in default ASCII format.

Example

Example: Reading the Calibration Number

The following program determines the number of times the 3245A has been calibrated.

```
10  OUTPUT 709;"CALNUM?"    !Calibration number query
20  ENTER 709;A             !Enter results
30  PRINT A                 !Print results
40  END
```

CALSTR/CALSTR?

Description

Store/Read Calibration Data. CALSTR stores calibration information in the CALibration RAM. **CALSTR?** returns the current calibration information stored. **CALSTR** functions only when the CAL jumper is ENABLED.

Syntax

CALSTR "[*string*]"

CALSTR?

Parameters

string Calibration information entered into the CALibration RAM. Up to 75 characters can be stored, and the string must be enclosed by apostrophies (') or quotation marks (").

Remarks

Storing Calibration Data (CALSTR)

After all calibration values have been entered (with the **CAL** or **CAL VOLTS** command), calibration string information such as calibration dates, temperature, etc. can be entered into the calibration RAM with the **CALSTR** command. Note that **CALSTR** functions only when the CAL jumper is in the ENABLE position.

Query Command (CALSTR?)

The **CALSTR?** command returns the current information stored in the calibration RAM by the **CALSTR** command.

Related Commands

CAL

Example

Example: Entering Calibration Information

This program uses **CALSTR** to enter calibration information into the CALibration RAM, and **CALSTR?** to recall the information. The program assumes that the CAL jumper is in the ENABLE position.

```
10   DIM A$[75]                                  !Dim array
20   OUTPUT 709;"CALSTR 'CALIBRATED 10/1/88 BY J.D.'" !Enter cal info
30   OUTPUT 709;"CALSTR?"                        !Read cal info
40   ENTER 709;A$                                !Enter cal info
50   PRINT A$                                    !Display cal info
60   END
```

A typical return is:

```
'CALIBRATED 10/1/88 BY J.D.'
```

CAT

Description

Catalog. Returns a catalog list of all user-defined arrays, variables and subroutines.

Syntax

CAT

Parameters

None.

Remarks

Data Returned

Each line returned by **CAT** contains information on one symbol. Array information is the array name, type, and number of elements. Variable information is the name and type. Subroutine information is the subroutine name, machine code size (bytes), and ASCII text size (bytes). The last line returned is the word DONE.

How DELSUB Affects CAT Listing

The **DELSUB** command deletes a specific subroutine from memory, but **CAT** still returns the subroutine name with code size = 0 and text size = 0. Since the name still appears in the catalog listing, the name cannot be redefined as something other than a subroutine.

How COMPRESS Affects CAT Listing

The **COMPRESS** command converts HP 3245A subroutines from ASCII to machine code format. After a **COMPRESS** command, **CAT** still returns the subroutine name with code size listed, but text size = 0. After **COMPRESS** is executed, subroutines cannot be listed (with **LIST**) or stepped (with **STEP**).

How SCRATCH Affects CAT Listing

SCRATCH deletes all user-defined arrays, variables, and stored subroutines from volatile memory. **SCRATCH** also removes all arrays, variables, and subroutines from the catalog listing.

Data Destination

When **CAT** is entered from the front panel, the response is displayed on the front panel display. When **CAT** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

Related Commands

COMPRESS, DELSUB, MEMAVAIL?, SCRATCH

Example

Example: Catalog Listing (CAT)

This program shows results returned by **CAT** for subroutine BEEPER, REAL variables I and R, memory), and INTEGER array A.

```
10   !file CAT
20   !
30   DIM Name$[60]                    !Dimension array
40   CLEAR 709                        !Clear HP 3245A
50   OUTPUT 709;"RST"                 !Reset HP 3245A
60   OUTPUT 709;"SCRATCH"             !Clear HP 3245A memory
70   OUTPUT 709;"USE 0"               !Use channel 0
80   !
90   OUTPUT 709;"SUB BEEPER"          !Begin sub BEEPER
100  OUTPUT 709;"  FOR I = 1 TO 5"    !Begin count loop
110  OUTPUT 709;"    BEEP ONCE"       !Beep once
120  OUTPUT 709;"    WAIT .5"         !Wait 0.5 seconds
130  OUTPUT 709;"  NEXT I"            !Increment count
140  OUTPUT 709;"SUBEND"              !End sub BEEPER
150  !
160  OUTPUT 709;"REAL R"              !Def REAL variable R
170  OUTPUT 709;"INTEGER A(20)"       !A is 21-element INTEGER array
180  OUTPUT 709;"CAT"                 !Request CAT listing
190  !
200  REPEAT                           !Loop until done
210    ENTER 709;Name$                !Enter line
220    PRINT Name$                    !Display line
230  UNTIL Name$ = "DONE"             !DONE is last word returned
240  END
```

A typical return follows. Note that I was automatically defined as a REAL variable since it was not specifically defined as an INTEGER variable.

```
IARRAY   A        SIZE     21
REAL     R
REAL     I
SUB      BEEPER   SIZE    268,  TEXT SIZE   136
DONE
```

CLEAR (DCL or SDC)

Description

CLEAR is the same as the HP 3245A **CLR** command and is executed immediately upon receipt (if **INBUF ON** is set). **CLEAR** (or **CLR**) does the following:

- Halts all commands or subroutines in progress.
- Clears all status register bits except bit 4 (READY).
- Clears any pending service request.
- Clears partially entered commands (HP-IB or front panel).
- Clears all stored error messages from error register.

- Clears the HP-IB input and output buffers.
- Enables the front panel display (if disabled by **DISP OFF**).
- Disables output memory mode (**MEM OFF**).
- Sets ASCII output format (**OFORMAT ASCII**).
- Sets monitor mode to **MON STATE CHANA** (**MON** command).

Syntax

CLEAR 7
CLEAR 709

Examples

```
CLEAR 7       !Immediately clear all devices (device clear) on bus
              !(select code 7).

CLEAR 709     !Immediately clear device (selected device clear) at
              !address 9 (select code 7).
```

Description

Device Clear. **CLR** places the HP 3245A in the same state as set by the front panel **Clear** key or by the HP-IB Device Clear or Selected Device Clear commands.

Syntax

CLR

Parameters

None.

Remarks

CLR Command Action

- Halts all commands or subroutines in progress.
- Clears all status register bits except bit 4 (READY).
- Clears any pending service request.
- Clears partially entered commands (HP-IB or front panel).
- Clears all stored error messages from error register.

- Clears the HP-IB input and output buffers.
- Enables front panel display (if disabled by **DISP OFF**).
- Disables output memory mode (**MEM OFF**).
- Sets ASCII output format (**OFORMAT ASCII**).
- Sets monitor mode to **MON STATE CHANA** (**MON** command).

Effect of CLR on Subroutines

If a subroutine is being entered when the **CLR** command is executed, the subroutine is scratched and the HP 3245A returns to the normal command mode. (The **CLR** command should not be used in a subroutine.)

CLR Command vs. Device Clear Messages

The **CLR** command is not recognized until all previously entered commands have been executed. However, the HP-IB Device Clear (e.g., **CLEAR 7**) and Selected Device Clear (e.g., **CLEAR 709**) commands abort any currently executing commands.

Related Commands

CIIL, CLROUT, GAL, RESET, RQS, RST

Example

Example: Using CLR, CLEAR, and CLEAR 709 Commands

The following program lines show the effect of the HP 3245A **CLR** command versus the HP-IB **CLEAR** and **CLEAR 709** commands on an HP 3245A at address 709.

```
10  OUTPUT 709;"CLR"     !Clear HP 3245A at address 09 when the
                         !command is executed.

10  CLEAR 7              !Immediately clear all devices (Device
                         !Clear) on bus (select code 7).

10  CLEAR 709            !Immediately clear device (Selected Device
                         !Clear) at address 9 (select code 7).
```

Description

Clear Output Buffer. Clears (erases) the data in the HP-IB output buffer. When the output buffer is enabled, **CLROUT** is useful to ensure that the next query from the controller gives a predictable response.

Syntax

CLROUT

Parameters

None.

Remarks

Using OUTBUF and INBUF with CLROUT

With **OUTBUF OFF** (output buffer disabled), it is usually not necessary to clear the output buffer since new data overwrites existing data in the buffer.

With **INBUF ON** (input buffer enabled), commands following the **CLROUT** command may be accepted and stored before **CLROUT** is executed. If the commands return data, the data returned may be old data, new data, or a combination of old and new data.

Related Commands

CLR, INBUF, OUTBUF

Example

Example: Clearing the HP-IB Output Buffer

This program statement clears the HP-IB output buffer.

```
          .
          .
100  OUTPUT 709;"CLROUT"    !Clear HP 3245A output buffer
          .
          .
```

COMPRESS

Description

Compress Subroutine. Converts the text of the named subroutine from ASCII to machine code. This saves space in memory, but compressed subroutines cannot be listed or stepped.

Syntax

COMPRESS *sub_name*

Parameters

sub_name Subroutine name. Subroutine names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore character ("_"), or the question mark ("?"). Subroutine names must not be the same as HP 3245A commands or parameters, previously defined array or variable names, or stored state names.

Remarks

Use COMPRESS Only After Subroutine is Tested

Use **COMPRESS** only after a subroutine has been debugged and tested, since a compressed subroutine cannot be listed (with **LIST**) or stepped (with **STEP**). Attempting to list or step a compressed subroutine does not produce an error but no action is taken. To avoid memory fragmentation, compress each debugged subroutine before downloading other subroutines. Compressed subroutines continue to appear in the catalog listing (**CAT**) with machine size (SIZE) given, but text size = 0 (TEXT SIZE 0).

Related Commands

CAT, LIST, STEP, SUB, SUBEND

Example

Example: Compressing a Subroutine (COMPRESS)

This program compares the effects of **COMPRESS** with **DELSUB** on HP 3245A subroutines. Subroutine V_OUTPUT is compressed with **COMPRESS**, while subroutine BEEPER is deleted with **DELSUB**. Note that V_OUTPUT can be run, but can no longer be listed or stepped. In contrast, BEEPER cannot be run although the subroutine name appears in the **CAT** list.

```
10   !file COMPRESS
20   !
30   DIM C$[60]                         !Dim cont array
40   CLEAR 709                          !Clear HP 3245A
50   OUTPUT 709;"RST"                   !Reset HP 3245A
60   OUTPUT 709;"SCRATCH"               !Clear HP 3245A memory
70   OUTPUT 709;"USE 0"                 !Use channel A
80   !
90   OUTPUT 709;"SUB V_OUTPUT"          !Begin sub V_OUTPUT
100  OUTPUT 709;"  DIM VOUT(3)"         !Dimension array
110  OUTPUT 709;"  FILL VOUT 1.1,2.2,3.3,4.4"  !Enter array values
120  OUTPUT 709;"  APPLY DCMEMV 4,VOUT"  !Output triggered DCV
130  OUTPUT 709;"  WAIT 3"              !Wait 3 seconds
140  OUTPUT 709;"  FOR I = 1 TO 3"      !Begin count loop
150  OUTPUT 709;"    TRIGIN SGL"        !Single trigger
160  OUTPUT 709;"    WAIT 3"            !Wait 3 seconds
```

```
170  OUTPUT 709;"  NEXT I"                !Increment loop
180  OUTPUT 709;"SUBEND"                  !End subroutine
190  OUTPUT 709;"COMPRESS V_OUTPUT"       !Compress sub V_OUTPUT
200  !
210  OUTPUT 709;"SUB BEEPER"              !Begin sub BEEPER
220  OUTPUT 709;"  FOR I = 1 TO 5"        !Begin count loop
230  OUTPUT 709;"    BEEP ONCE"           !Beep once
240  OUTPUT 709;"    WAIT .5"             !Wait 0.5 seconds
250  OUTPUT 709;"  NEXT I"                !Increment count
260  OUTPUT 709;"SUBEND"                  !End sub BEEPER
270  OUTPUT 709;"DELSUB BEEPER"           !Delete sub BEEPER
280  !
290  OUTPUT 709;"CAT"                     !Catalog listing
300  !
310  REPEAT                               !Repeat until DONE
320    ENTER 709;C$                       !Enter line
330    PRINT C$                           !Display line
340 UNTIL C$="DONE"                       !DONE is last word
350 END
```

A typical return follows. Since subroutine BEEPER was deleted from memory (SIZE 0, TEXT SIZE 0) by the **DELSUB** command, the subroutine cannot be run, but the subroutine name BEEPER still appears in the **CAT** list. The sub V_OUTPUT was compressed (SIZE 462, TEXT SIZE 0) by the **COMPRESS** command, so the subroutine can still be run but cannot be stepped or listed. Also, note that the size of REAL array VOUT has been changed to 0 elements by the **COMPRESS** command.

```
SUB      BEEPER      SIZE       0, TEXT SIZE      0
REAL     I
RARRAY   VOUT        SIZE       0
SUB      V_OUTPUT    SIZE     462, TEXT SIZE      0
DONE
```

CONT

Description

Continue Subroutine. Continues execution of a paused or stepped RUN subroutine without further intervention.

Syntax

CONT

Parameters

None.

Remarks

CONT Resumes Execution of Current Subroutine

CONT resumes execution of a RUN subroutine currently paused or in single-step mode. When executed, **CONT** allows the subroutine to execute to completion, starting with the command after **PAUSE** or the last command to be stepped.

Related Commands

PAUSE, RUN, STEP, SUB, SUBEND

Example

Example: Continuing a Paused Subroutine (CONT)

In this program, the **0** key is redefined as the **CONT** command to continue subroutine V_OUTPUT when key is pressed. When the program executes, the output from channel A is set to 1.1 VDC and remains at that level until the **0** key and **Enter** keys are pressed. (**CONT** is displayed on the front panel when the **0** key is pressed.) Each time the **0** and **Enter** keys are pressed, the subroutine is continued and the output is increased 1.1 VDC (2.2 VDC, 3.3 VDC, 4.4 VDC).

```
10   !file CONT
20   !
30   CLEAR 709                              !Clear HP 3245A
40   OUTPUT 709;"RST"                       !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"                   !Clear HP 3245A memory
60   OUTPUT 709;"USE 0"                     !Use channel A
70   OUTPUT 709;"DEFKEY 0, 'CONT'"          !Define key 0 as "CONT"
80   !
90   OUTPUT 709;"SUB V_OUTPUT"              !Begin subroutine
100  OUTPUT 709;"  DIM VOUT(3)"             !Dimension array
110  OUTPUT 709;"  FILL VOUT 1.1,2.2,3.3,4.4"  !Enter array values
120  OUTPUT 709;"  APPLY DCMEMV 4,VOUT"     !Output triggered DCV
130  OUTPUT 709;"  FOR I = 1 TO 3"          !Begin count loop
140  OUTPUT 709;"    PAUSE"                 !Pause subroutine
150  OUTPUT 709;"    TRIGIN SGL"            !Single trigger
160  OUTPUT 709;"  NEXT I"                  !Increment loop
170  OUTPUT 709;"SUBEND"                    !End subroutine
180  !
190  OUTPUT 709;"RUN V_OUTPUT"              !Run subroutine
200  OUTPUT 709;"LOCAL"                     !Front panel mode
210  END
```

Description

Cosine. Returns the cosine of the angle (in radians) represented by the argument.

Syntax

COS (*argument*)

Parameters

argument

The *argument* parameter must be a number or numeric expression in radians. The valid range is ± 2.98156824429204 E+8.

Remarks

Related Commands

ATN, SIN

Example

Example: Using the COS Function

This program computes the cosine of 30 degrees (0.5235 radians) and displays the result (.86607479) on the controller CRT.

```
10  OUTPUT 709;"VREAD COS(.5235)"   !Compute and read COS of 30 degrees
20  ENTER 709;A                     !Enter result
30  PRINT "Cosine = ";A             !Display result
40  END
```

A typical return is:

```
    Cosine = .86607479
```

CRESET

Description

Channel Reset. Places the specified channel in its reset state. This command performs the same action as the **RESET** *ch* (or **RST** *ch*) command.

Syntax

CRESET *ch*

Parameters

ch Channel to be reset. **CRESET 0** or **CRESET CHANA** resets channel A while **CRESET 100** or **CRESET CHANB** resets channel B.

Remarks

CRESET Same as RESET for a Channel

The actions for **CRESET** are the same as for **RESET** *ch* on the channel specified. Refer to the **RESET** command for details.

Example

Example: Reset HP 3245A Channels

Line 100 resets channel A and line 110 resets channel B.

```
          .
          .
100  OUTPUT 709;"CRESET 0"       !Reset channel A
110  OUTPUT 709;"CRESET CHANB"   !Reset channel B
          .
          .
```

CTYPE? (CTYPE)

Description

Channel Type Query. CTYPE? (or **CTYPE**) returns a number indicating whether or not the specified channel is installed. **CTYPE?** returns "21" if the channel is installed or "0" if the channel is not installed.

Syntax

CTYPE? *channel*

Parameters

channel Channel which is queried. CHANA or 0 specifies channel A, CHANB or 100 specifies channel B.

Remarks

Data Returned

CTYPE? (or **CTYPE**) returns "21" if the specified channel is installed, or returns "0" if the channel is not installed.

Data Destination

If the **MEM** command is used, the value returned by **CTYPE?** is stored in the specified variable or array. When **CTYPE?** is executed from the front panel, the response is displayed on the front panel display. When **CTYPE?** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

HP-IB Data Format

The HP 3245A returns numeric results in either ASCII or binary format (see the **OFORMAT** command). In the default ASCII format, **CTYPE?** returns integer results in 6-digit signed notation. In the binary format, **CTYPE?** returns integer results in 16-bit, 2's complement notation.

Related Commands

ID?, IDN?

Example

Example: Determining if a Channel is Installed

This program returns a number indicating whether the specified channel is installed in the 3245A.

```
10  OUTPUT 709;"CTYPE? CHANB"          !Query if channel B is installed
20  ENTER 709;A                        !Return number
30  PRINT A                            !Display number
40  END
```

If channel B is installed, the number returned is:

```
    21
```

DCOFF/DCOFF?

Description

Set/Read DC Offset Voltage. **DCOFF** sets the DC offset voltage for sine, ramp, square, and arbitrary waveform outputs on the **USE** channel. **DCOFF?** returns the DC offset value on the **USE** channel.

Syntax

DCOFF *volts*

DCOFF?

Parameters

volts Offset voltage in DC volts. Positive or negative number such that the peak AC magnitude of the output waveform, plus the magnitude of the offset, does not exceed 50% of the selected peak-to-peak range. Power-on/reset *volts* = 0.

Remarks

Using DCOFF

If the autorange mode is enabled (**ARANGE ON** or **RANGE AUTO**) when the **DCOFF** command is executed, the HP 3245A automatically changes range as required to accommodate the offset voltage. With the 0Ω mode (**IMP 0**), the actual DC offset voltage equals the programmed value with or without load termination. In the 50Ω mode (**IMP 50**), the actual offset voltage equals the programmed value ONLY when the channel is terminated with a 50Ω load.

OUT OF RANGE Error

If the present AC amplitude for a specified DC offset is greater than the maximum allowed for a new DC offset, selecting the new DC offset will generate an "OUT OF RANGE" error. For example, if channel A is set for 50Ω impedance, 5 V ac PP output, and 2.5 V DC offset, attempting to set **DCOFF 2.6** will generate the error "OUT OF RANGE --- 5.000000E+0", where the error value refers to the amplitude rather than the DC offset.

Query Command (DCOFF?)

The **DCOFF?** command returns the DC offset voltage setting on the **USE** channel (channel A or B).

Related Commands

APPLYs, ARANGE, IMP, RANGE, USE

Example

Example: Ramp Waveform Output With DC Offset (DCOFF)

This program outputs a 5 Vac PP ramp waveform with 2.5 VDC offset.

```
10    !file DCOFF
20    !
30    CLEAR 709                    !Clear HP 3245A
40    OUTPUT 709;"RST"             !Reset HP 3245A
50    OUTPUT 709;"SCRATCH"         !Clear HP 3245A memory
60    !
70    OUTPUT 709;"USE 0"           !Use channel A
80    OUTPUT 709;"APPLY RPV 5"     !Output ramp waveform @ 5 Vac PP
90    OUTPUT 709;"DCOFF 2.5"       !Set offset = 2.5 VDC
100   END
```

DCRES/DCRES?

Description

Set/Read DC Resolution. **DCRES** selects high-resolution or low-resolution mode for DC voltage and current outputs on the **USE** channel. **DCRES?** returns the setting (HIGH or LOW) on the **USE** channel.

Syntax

DCRES *mode*

DCRES?

Parameters

mode The *mode* parameters follow. Power-on/reset *mode* = HIGH.

mode	Definition
HIGH	High-Resolution Mode: 40 msec settling time (at 0.0001% resolution); 6 digits of resolution; 2 DC voltage ranges; 4 DC current ranges.
LOW	Low-Resolution Mode: 100 µsec settling time (at 0.05% resolution); 3 digits of resolution; 7 DC voltage ranges; 4 DC current ranges.

Remarks

Resolution Modes (DCRES)

For greatest accuracy, use the high-resolution mode (**DCRES HIGH**) which provides six digits of resolution. For decreased settling time with lower (three-digit) resolution, use the low-resolution mode (**DCRES LOW**).

The high-resolution mode has two DC voltage ranges: 1 VDC and 10 VDC. The low-resolution mode has seven DC voltage ranges from less than 200 mVDC up to 10 VDC (refer to the **APPLY DCV** command). Four DC output current ranges are available from 0.1 mA to 100 mA for both high-resolution and low-resolution modes (refer to the **APPLY DCI** command).

OUT OF RANGE Error

If the preset DC output in **DCRES HIGH** mode is greater than the maximum allowed in **DCRES LOW**, selecting **DCRES LOW** will generate an "OUT OF RANGE" error. For example, if you set channel A to output 10.1 VDC with **DCRES HIGH**, selecting **DCRES LOW** will generate the error "OUT OF RANGE ----1.010000E+1". Note that the error value refers to the amplitude rather than to the DC resolution.

Query Command (DCRES?)

DCRES? returns the setting (HIGH or LOW) on the **USE** channel (channel A or B).

Related Commands

APPLY DCI, APPLY DCV, DCRES?, USE

Example

Example: Using the DCRES Command (DCRES)

When this program executes, **RST** sets high-resolution mode (**DCRES HIGH**). Then, **DCRES LOW** sets low-resolution mode on channel A. Since the 1.25V range and 0Ω output impedance are used, the resolution of the 1.123 VDC output is 625 mV (refer to the **APPLY DCV** command).

```
10   !file DCRES
20   !
30   CLEAR 709                        !Clear HP 3245A
40   OUTPUT 709;"RST"                 !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"             !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"               !Use channel A
80   OUTPUT 709;"  DCRES LOW"         !Set low-resolution mode
90   OUTPUT 709;"  APPLY DCV 1.123"   !Output 1.123 DCV
100  END
```

DEFKEY/DEFKEY?

Description

Define Key/Read Defined Key. DEFKEY allows a user-defined string to be assigned to a NUMERIC key on the HP 3245A front panel. **DEFKEY?** returns the current definition for the specified NUMERIC key.

Syntax

DEFKEY *key*, *string*

DEFKEY? *key*

Parameters

key The *key* parameter can be any of the front panel NUMERIC keys (0 through 9).

string The *string* parameter (up to 75 characters) is assigned to the specified NUMERIC *keys*. When **DEFKEY** is executed from the front panel, the *string* parameter must be enclosed in quotation marks ("*string*").

When **DEFKEY** is executed from the controller, the *string* parameter must be enclosed in double quotation marks (""*string* "") or apostrophes ('*string*'). Use the empty string (e.g., `OUTPUT 709;"DEFKEY 1 ''"`) to return a NUMERIC key to its default definition. Use **DEFKEY DEFAULT** to return all NUMERIC keys to the default definition.

Remarks

Redefining NUMERIC Keys (DEFKEY)

Redefined NUMERIC keys may contain complete commands or command fragments. Multiple commands may be stored in a single key when separated by a semicolon (;). Up to 75 characters can be stored in a NUMERIC key string. Only the NUMERIC keys (0 through 9) can be redefined. The front panel FUNCTION/RANGE, MENU, and USER keys cannot be redefined. Strings defined by the **DEFKEY** command are stored in continuous memory and are retained when power is removed from the HP 3245A.

Query Command (DEFKEY?)

The **DEFKEY?** *key* command returns the current definition of the specified key.

Related Commands

None.

Example

Example: Redefining a NUMERIC Key (DEFKEY)

This program redefines NUMERIC key 5 with the string "`USE 0;APPLY DCV 0.1`" and returns the redefined function to the controller CRT. When NUMERIC key **5** is pressed, the HP 3245A outputs 0.1 DCV from the **USE** channel. To return key **5** to its original definition, use `OUTPUT 709;"DEFKEY 5 ''"`. To return ALL NUMERIC keys to their original definitions, use **OUTPUT 709;"DEFKEY DEFAULT"**. Note that cycling power does NOT return the NUMERIC keys to their default definitions.

```
10   !file DEFKEY
20   !
30   DIM A$[160]                      !Dim controller array
40   CLEAR 709                        !Clear HP 3245A
50   OUTPUT 709;"RST"                 !Reset the 3245A
60   OUTPUT 709;"SCRATCH"             !Clear 3245A memory
70   !
80   OUTPUT 709;"DEFKEY 5,'USE 0;APPLY DCV 0.1'"  !Redefine key 5
90   OUTPUT 709;"DEFKEY? 5"           !Read key 5 new definition
100  ENTER 709;A$                     !Enter definition
110  PRINT A$[2;LEN(A$)-2]            !Display definition
120  OUTPUT 709;"LOCAL"               !Place 3245A in local state
130  END
```

A typical return is:

```
USE 0;APPLY DCV 0.1
```

DELAY/DELAY?

Description

Set/Read Output Delay. **DELAY** specifies a time interval during which outputs from the **USE** channel are allowed to settle before the next command is executed. **DELAY?** returns the delay value set for the **USE** channel.

Syntax

DELAY [*time*]

DELAY?

Parameters

time Specifies settling time (range = 0 to 9.99 seconds) with 0.01 second resolution. Power-on/reset *time* = 0.04 seconds. Default *time* = automatically determined by function, range, DC resolution, and output impedance.

Remarks

Default Delays

If a delay is not specified, the HP 3245A determines a default delay based on present function, range, DC resolution, and output impedance. If the output function, range, resolution, or output impedance are changed without specifying a new delay, the default delay time is updated based on the new selection.

Once a delay time is set with **DELAY**, the value does not change until another value is specified or the **USE** channel is changed. A delay time can be set which is shorter than the default value, but the resultant settling time may be too short to allow the channel to meet accuracy specifications for that output function.

Change in:	Default Delay
Function:	
High-res DCV/DCI from other function.	1 sec
All other function changes.	30 msec
Output Value:	
High-resolution mode	40 msec
Low-resolution mode (same range)	0 msec
DC triggered-seq mode (first value)	30 msec
DC trig-seq mode (after first value)	0 msec
All other modes	30 msec
DC Offset:	
Changes default delay for sine, ramp, square, and arb functions	30 msec
Range:	
High-resolution mode	40 msec
All other modes	30 msec
Output Impedance (voltage only):	
High-resolution mode	40 msec
All other modes	30 msec

Commands Affected by DELAY

The **DELAY** command affects commands which control the output function (**APPLY** commands); output range (**ARANGE** and **RANGE**); DC resolution (**DCRES**); and output impedance (**IMP**). When the **USE** channel is operating in the DC triggered-sequence mode (**APPLY DCMEMI** or **APPLY DCMEMV**), only the first value in the sequence is delayed.

Query Command (DELAY?)

DELAY? returns the delay value set for the **USE** channel (channel A or B).

Related Commands

APPLYs, DCRES, IMP, RANGE, USE

Example

Example: Specifying a Delay Time (DELAY)

This program sets a delay between the output of two DC voltages. **APPLY DCV 5** (line 90) outputs 5 VDC. After a two-second delay, **APPLY DCV 2.5** (line 100) outputs 2.5 VDC.

```
10   !file DELAY
20   !
30   CLEAR 709                     !Clear HP 3245A
40   OUTPUT 709;"RST"              !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"          !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"            !Use channel A
80   OUTPUT 709;"  DELAY 2"        !Set 2 second delay
90   OUTPUT 709;"  APPLY DCV 5"    !Output 5 VDC for 2 seconds
100  OUTPUT 709;"  APPLY DCV 2.5"  !Output 2.5 VDC
110  END
```

DELSUB

Description

Delete Subroutine. Deletes the specified subroutine from HP 3245A memory but does not delete the subroutine name from the catalog listing (contrast with **SCRATCH**).

Syntax

DELSUB *sub_name*

Parameters

sub_name

Subroutine name. Subroutine names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore character ("_"), or the question mark ("?"). Subroutine names must not be the same as HP 3245A commands or parameters, previously defined array or variable names, or stored state names.

Remarks

Subroutine Name Not Deleted

Deleting a subroutine recovers the memory space and the subroutine cannot be called (with **CALL**) or run (with **RUN**). The deleted subroutine name still appears in the **CAT** listing (with size zero) and cannot be redefined as anything other than a subroutine. Use **SCRATCH** to remove definitions of all user-defined names.

Related Commands

CAT, PURGE, SCRATCH

Example

Example: Deleting a Subroutine

Refer to the **COMPRESS** command for an example program which compares the effects of **COMPRESS** with **DELSUB** on HP 3245A subroutines.

Description

Demonstation Waveforms. Demonstates some arbitrary waveform capabilities of the 3245A.

Syntax

DEMO *ALL*
HEART
HP3245
NOISE

Parameters

ALL Using **DEMO ALL** continuously cycles through the three demonstration waveforms: **HEART**, **HP3245**, and **NOISE**. All outputs are from the front panel Output connectors.

HEART **DEMO HEART** generates a 0.25 V ac PP electrocardiogram waveform on channel A and a 2.0 V ac PP phonocardiogram waveform (suitable for directly driving 8Ω speakers) on channel B.

HP3245 **DEMO HP3245** generates a 0.1 V ac PP "HP3245" logo on channels A and B.

NOISE **DEMO NOISE** generates a 0.4 V ac PP noisy sinewave on channels A and B.

Remarks

None.

Example

This program generates a 0.25 V ac PP electrocardiogram waveform on channel A and a 2.0 V ac PP phonocardiogram waveform on channel B.

```
10   CLEAR 709                 !Clear HP 3245A
20   OUTPUT 709;"RST"          !Reset HP 3245A
30   OUTPUT 709;"SCRATCH"      !Clear HP 3245A memory
40   OUTPUT 709;"  DEMO HEART" !Output EKG and phonocardiogram
```

DIM

Description

Dimension REAL Array. Reserves space in HP 3245A memory to store REAL numeric arrays.

Syntax

DIM *array_name*(*max_index*) [,*array_name* (*max_index*) , ...]

Parameters

array_name

Array name. Array names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore character ("_"), or the question mark ("?"). Array names must not be the same as HP 3245A commands or parameters, previously defined array or variable names, or stored state names.

max_index

Specifies the number of elements in the array. Valid range of *max_index* is 0 to 32767. The lower bound (also known as option base) for all arrays is zero so the number of elements in the array is one more than *max_index*. For example, "DIM A(5)" reserves memory space for a REAL array A with six elements (0 through 5).

Remarks

Redimensioning Arrays (DIM)

Executing **DIM** (or **REAL**) sets up a REAL array and fills all elements with zeroes. Arrays may be redimensioned using **DIM** or **REAL**. A redimensioned array is filled with zeroes in all elements. If the array is redimensioned to the same size, it is zeroed and all previous data is lost. Any number of arrays may be dimensioned up to the limit of available HP 3245A memory.

User-Defined Arrays are Global

All REAL numeric arrays are global among all front panel, HP-IB, and subroutine operations. A user-defined array may be used in any HP 3245A command where a numeric parameter is required.

DIM Command Stored in Subroutine

If a **DIM** command is stored in a subroutine, the array name is defined immediately, but data storage for the array is not allocated until the subroutine is run or called (the name cannot be used for a subroutine name, another array name, or a variable name).

DIM Command Versus REAL Command

The **DIM** command defines REAL arrays only, while the **REAL** command defines REAL arrays and REAL variables.

Recovering Memory Space

Use the **SCRATCH** command to recover memory space allocated for arrays by the **DIM** command. After **SCRATCH**, any valid name can be used.

Related Commands

DISP, FETCH, FILL, INTEGER, REAL, SIZE?, VREAD

Example

Example: Dimensioning REAL Array

This program line sets up REAL array A with 11 elements (0 through 10).

```
            .
            .
100  OUTPUT 709;"DIM A(10)"    !Define REAL array A with 11 elements
            .
            .
```

DISP (DSP)/DISP? (DSP?)

Description

Enable/Read Front Panel Display. **DISP** (or **DSP**) enables or disables the front panel display. This command can also be used to display a string, the contents of a variable, a number, or a numeric expression. **DISP?** (or **DSP?**) returns a quoted string containing the contents of the front panel display.

Syntax

DISP *mode*

or

DISP [**MSG**] [*message*]

DISP?

Parameters

mode Enables or disables the front panel display. The parameters follow. Power-on/reset *mode* = ON.

mode	Description
ON	Front panel display enabled.
OFF	Front panel display disabled (arrow/dashes displayed).

MSG MSG is an optional parameter used to indicate to the HP 3245A that a message is to be displayed on the front panel.

message The *message* may be a quoted string of characters (quotation marks are not displayed), the value of a variable, a number, or a numeric expression (enclosed in parentheses).

If **DISP** is executed from the front panel, the *message* parameter must be enclosed in quotation marks ("*message*"). If **DISP** is executed from the controller, *message* must be enclosed in double quotation marks (" "*message*" ") or apostrophes ('*message*'). When using the controller, the message can be entered in either upper case or lower case letters.

Remarks

Disabling/Clearing Front Panel Display (DISP)

When disabled (**DISP OFF**), the front panel displays all dashes and all annunciators are OFF. The display can be reenabled with **DISP ON** or by pressing any front panel key which changes the display. With the display disabled, command execution speed increases since the HP 3245A does not continually update the front panel display. To clear the front panel display, send "**DISP** ' ' ".

Query Command (DISP?)

DISP? (or **DSP?**) returns a quoted string containing the contents of the front panel display, including all characters outside the 15-character display window (up to 256 characters). When **DISP?** is executed from the front panel, "DISP?" is displayed on the front panel. When **DISP?** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

DISP? is useful for reading **MON** displays. The **MON** command normally displays results on the front panel. However, you can use **DISP?** to send the results to the controller.

Related Commands

FETCH, VREAD

Examples

Example: Display Message and Variable

This program displays the cosine of 30 degrees (.5235 radians) on the front panel display.

```
10  OUTPUT 709;"VAL = COS(.5235)"        !Compute cosine
20  OUTPUT 709;"DISP 'Cosine = ',VAL"    !Display result
30  END
```

Example: Reading MON Messages

This program monitors the state of channel A and displays the results on the controller CRT. (Note that the state of channel A is also displayed on the front panel display.)

```
10  DIM Message$[256]          !Dimension string array
20  OUTPUT 709;"RESET"         !Reset HP 3245A
30  OUTPUT 709;"MON STATE 0"   !Display channel A state
40  OUTPUT 709;"DISP '' "      !Clear front panel display
50  WAIT 1                     !Wait for state on display
60  OUTPUT 709;"DISP?"         !Read front panel display
70  ENTER 709;Message$         !Enter message
80  PRINT Message$             !Print message
90  END
```

Since **RESET** sets channel A to its power-on condition, a typical return is:

```
"0.000000E+0DCV, FREQ 1000.000 1000.000,
 DCOFF 0.000000E+00,DUTY 50.0,PANG 0.000,RANGE 1.0,ARANGE ON,
 TERM FRONT,IMP 0,DCRES HIGH,TRIGMODE OFF,TRIGIN HIGH, TRIGOUT
 OFF,SYNCOUT OFF,REFIN INT,REFOUT EXT,DELAY 0.04"
```

DIV

Description

Division. Returns the integer portion of a division. Normal division takes place but all digits to the right of the decimal point are truncated (contrast with **MOD**).

Syntax

numerator **DIV** *denominator*

Parameters

numerator

The *numerator* parameter may be a number, numeric variable, math function, array element, or numeric expression (enclosed in parentheses).

denominator

The *denominator* parameter must be a value NOT equal to zero. This parameter may be a number, numeric variable, math function, array element, or numeric expression.

Remarks

Related Commands

MOD, INT

Example

Example: Using the DIV Function

This program divides 7 by 3 and displays the integer portion of the division (2) on the controller CRT.

```
10  OUTPUT 709;"VREAD 7 DIV 3"   !Compute and read DIV value
20  ENTER 709;A                  !Enter result
30  PRINT "DIV = ";A             !Print result
40  END
```

A typical return is:

```
  DIV = 2
```

DRIVETBn/DRIVETBn?

Description

Set/Read Trigger Bus Source. When **TRIGIN TB0/TB1** is set, **DRIVETBn** selects the source to drive the specified trigger bus (0 or 1). **DRIVETBn?** returns the source to drive the specified trigger bus.

Syntax

DRIVETB*n* *source*

DRIVETB*n*?

Parameters

n The *n* parameter specifies the trigger bus to be used. A "0" specifies backplane trigger bus 0, while a "1" specifies backplane trigger bus 1.

source Source which drives the specified trigger bus. The *source* parameters follow, where LOW = 0 V and HIGH = +5 V. Power-on/reset *source* = OFF.

source	Definition
OFF	Disable the trigger bus drivers.
LOW	Force the trigger bus low (0 Volts).
HIGH	Force the trigger bus high (+5 Volts).
SGL	Pulse the trigger bus (LOW then HIGH) when the command is executed.
TRG	Pulse the trigger bus (LOW then HIGH) when the HP 3245A receives a TRG command or an HP-IB Group Execute Trigger.

Remarks

Using DRIVETBn OFF

When using the external trigger input ports (TB0 and TB1) on the rear panel to input an external trigger, set **DRIVETBn OFF** to disable the trigger bus drivers. This allows TB0 and/or TB1 to act as input terminals.

REFOUT/SYNCOUT/DRIVETBn Interaction

Using **REFOUT TBn** or **SYNCOUT TBn** automatically resets **DRIVETBn** to **OFF**.

Query Command (DRIVETBn?)

The **DRIVETBn?** command returns the source setting (OFF, LOW, HIGH, or TRG) for the specified trigger bus.

Related Commands

REFIN, REFOUT, SYNCOUT, TBn?, TRIGIN

Example

Example: Driving Trigger Bus 0 (DRIVETB)

This program drives backplane trigger bus 0 LOW, then HIGH, when **DRIVETB0 SGL** is executed. **DRIVETB0?** returns the source for TB0 (SGL) for the **DRIVETB0** command.

```
10   !FILE DRIVETB
20   !
30   CLEAR 709                       !Clear HP 3245A
40   OUTPUT 709;"RST"                !Reset HP 3245A
60   OUTPUT 709;"SCRATCH"            !Clear HP 3245A memory
60   OUTPUT 709;"USE 0"
70   !
80   OUTPUT 709;"TRIGIN TB0"         !Set trigger source
90   OUTPUT 709;"DRIVETB0 SGL"       !Pulse TB0 LOW then HIGH
100  OUTPUT 709;"DRIVETB0?"          !Query TB0 source
110  ENTER 709;A$                    !Enter source
120  PRINT "TB0 DRIVE SOURCE IS";A$  !Display source
130  END
```

A typical return is:

```
TB0 DRIVE SOURCE IS SGL
```

DTEST (DTST)

Description

Data Test. **DTEST** (or **DTST**) performs a data test on channels A and/or B and returns the measured values. Both channels can be tested at the same time. Note that the data test is also performed during an FTEST.

Syntax

DTEST [*ch*, [*ch*]]

Parameters

ch Channel parameter. *ch* = 0 (channel A), = 100 (channel B), = 1 (channel A rear panel), = 101 (channel B rear panel). If no channel is specified, channels A and B are tested. Note that **DTEST** resets the specified channel(s).

Remarks

Data Returned

DTEST performs the same test as the **FTEST** command for each channel. However, **DTEST** returns the actual measurements (175 per channel.

Cabling Required for Data Test

To perform the data test, the Output port (front or rear panel) of the channel being tested must be connected to the channel's Trigger (I/O) port.

Data Destination

If the **MEM** command is used, the values returned by the **DTEST** command are stored in the specified array. When **DTEST** is executed from the front panel, the response is displayed on the front panel display. When **DTEST** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

HP-IB Data Format

The 3245A returns numeric results in either ASCII or binary format (see the OFORMAT command). In the default ASCII format, the **DTEST** command returns real results in 8-digit scientific notation. In the binary format, the **DTEST** command returns real results in IEEE-754 64-bit notation.

Related Commands

FTEST

Example

Example: Data Test on Channels A and B

This program performs a data test on channels A and B. The results of the test are stored in HP 3245A memory with the MEM command, are then read from memory, entered into the controller, and displayed.

```
10   DIM Test1(0:349)
20   OUTPUT 709;"DIM RESULTS(349)"
30   OUTPUT 709;"MEM RESULTS"
40   OUTPUT 709;"DTEST"
50   OUTPUT 709;"MEM OFF"
60   OUTPUT 709;"VREAD RESULTS"
70   ENTER 709;Test1(*)
80   PRINT Test1(*)
90   END
```

Should the HP 3245A fail a data test, return the instrument to your nearest Hewlett-Packard Sales and Support Office for repair.

DUTY/DUTY?

Description

Set/Read Duty Cycle. **DUTY** sets the duty cycle for ramp and square wave output waveforms. **DUTY?** returns the duty cycle on the **USE** channel.

Syntax

DUTY *%_duty*

DUTY?

Parameters

%_duty

Duty cycle between 5% and 95%. Power-on/reset *%_duty* = 50%.

Remarks

DUTY Valid for Waveforms up to 100 kHz

At power-on, the duty cycle for ramp and square waveforms is 50%. The duty cycle can be varied from 5% to 95% for frequencies up to 100 kHz. For square waveforms above 100 kHz, an error is generated if the duty cycle is set to a value other than 50% (**DUTY 50**).

Momentary Irregularities in the Output Waveform

Repeated specification of the same duty cycle (e.g., **DUTY 15;DUTY 15**) will cause momentary (approximately 120 msec) irregularities in the waveform because the waveform is reloaded with a new duty cycle on each execution.

Query Command (DUTY?)

The **DUTY?** command returns the current duty cycle on the **USE** channel (channel A or B).

Related Commands

APPLY RPV, APPLY SQV, FREQ, USE

Example

Example: Setting Duty Cycle (DUTY)

This program outputs a 5 Vac PP, 5 kHz, 25% duty cycle square waveform from channel A.

```
10   !file DUTY
20   !
30   CLEAR 709                   !Clear HP 3245A
40   OUTPUT 709;"RST"            !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"        !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"          !Use channel A
80   OUTPUT 709;"DUTY 25"        !Set 25% duty cycle
90   OUTPUT 709;"FREQ 5E3"       !Set frequency to 5 kHz
100  OUTPUT 709;"APPLY SQV 5"    !Output sq wave @ 5 Vac PP
110  END
```

ECHO

Description

Echo. Tests communication between the HP 3245A and the controller by sending a string to the HP 3245A and having it "echo" back.

Syntax

ECHO *string*

Parameters

string

Any set of allowable ASCII characters enclosed in either double quotation marks (" "*string*" ") or single apostrophes ('*string*').

Remarks

None.

Example

Example: Echoing a Message

This program sends a message from the controller to the HP 3245A which returns the message to the controller for display.

```
10  DIM A$[100]
20  OUTPUT 709;"ECHO 'HP 3245A Source' "
30  ENTER 709;A$
40  PRINT A$
50  END
```

A typical return is:

```
    'HP 3245A Source'
```

Description

End. Suppresses or asserts the HP-IB *End Or Identify* (EOI) signal. When enabled, the HP-IB EOI line is set true concurrent with the transmission of the last data byte for any output format.

Syntax

END *control*

Parameters

control

The *control* parameters follow. Power-on/reset *control* = OFF.

control	Definition
OFF	EOI line is suppressed.
ON	EOI line true concurrent with last data byte.
ALWAYS	Same action as ON.

Remarks

Command Termination

Normally, data messages over the HP-IB bus are sent using standard ASCII codes and are terminated by the ASCII carriage return/line feed (*cr lf*). Some controllers, however, cannot terminate data input on *cr lf*. For this reason, the EOI line in the HP-IB interface may be used to mark the end of the data message using the **END** command.

Using the EOI Line

With **END ON** set, the HP 3245A sets the EOI line true concurrent with the last byte of the data message (the line feed). With **END OFF** set, EOI is suppressed and the line feed terminates the message when the preceding data satisfies the controller variables.

Using END With the Output Buffer

With the output buffer OFF (**OUTBUF OFF**) (power-on), data sent to the buffer overwrites the data currently in the buffer. Thus, asserting EOI with the output buffer OFF is effective only for controllers which do not recognize *lf* as a terminator.

With the output buffer ON (**OUTBUF ON**), data from multiple commands is appended to data currently in the buffer. With **END ON**, EOI is asserted with the last byte (*lf*) of each command's data so multiple ENTER statements are required. However, with **END OFF**, EOI is suppressed and a single ENTER statement can be used.

Examples

Example: Suppressing EOI (END)

This program demonstrates the effect of **END OFF** (the power-on setting). In the program, the output buffer is enabled so that data will be appended (**DCRES?**, **RANGE?**). Because EOI is suppressed, a single ENTER statement enters the entire contents of the buffer into the controller. The line feed associated with the results of **RANGE?** terminates the data message.

```
10   !file END
20   !This program demonstrates the effect of END OFF (power-on).
30   !
40   CLEAR 709                    !Clear HP 3245A
50   OUTPUT 709;"RST"             !Reset HP 3245A
60   OUTPUT 709;"SCRATCH"         !Clear HP 3245A memory
70   !
80   OUTPUT 709;"OUTBUF ON"       !Set output buffer ON
90   OUTPUT 709;"USE 0"           !Use channel A
100  OUTPUT 709;"  DCRES?"        !Query DC resolution
110  OUTPUT 709;"  RANGE?"        !Query DC range
120  ENTER 709;A$,B               !Enter DC resolution
130  PRINT A$,B                   !Display resolution, range
140  END
```

A typical return is:

```
HIGH   1
```

Example: Asserting EOI (END_1)

This example is similar to the first, however; **END ON** is set. The output buffer is enabled such that the data returned by **DCRES? RANGE?** is appended. Since **END ON** is set, EOI is asserted with the last byte (line feed) of the DCRES? and with the last byte (line feed) of the RANGE? data. Therefore, two ENTER statements are required.

```
10   !file END_1
20   !This program demonstrates the effect of END ON.
30   !
40   CLEAR 709                    !Clear HP 3245A
50   OUTPUT 709;"RST"             !Reset HP 3245A
60   OUTPUT 709;"SCRATCH"         !Clear HP 3245A memory
70   !
80   OUTPUT 709;"OUTBUF ON"       !Set output buffer ON
90   OUTPUT 709;"END ON"          !Enable EOI to be asserted
100  OUTPUT 709;"USE 0"           !Use channel A
110  OUTPUT 709;"  DCRES?"        !Query DC resolution
120  OUTPUT 709;"  RANGE?"        !Query DC range
130  ENTER 709;A$                 !Enter DC resolution
140  ENTER 709;B                  !Enter DC range
150  PRINT A$,B                   !Display resolution, range
160  END
```

A typical return is:

```
HIGH   1
```

END IF

See the IF. . .END IF Command.

END WHILE

See the WHILE...END WHILE Command.

ERR?

Description

Error Query. Returns the error code of the most recent error, deletes that error code from the error register, and clears the status register error bit after all errors are read. Note that **ERR?** cannot be executed from the front panel.

Syntax

ERR?

Parameters

None.

Remarks

Data Returned

One integer is returned corresponding to the most recent error in the error register. Errors are read in a first-in-first-out fashion (i.e., the first error to occur is the first error to be read). If no error codes are in the error register, "0" is returned. Up to four errors can be stored in the error register.

ERR? Clears the Error Register

ERR? reads the error messages and clears the error register one error at a time. Errors that occur while the register is full are displayed as usual, but are not stored in the error register. Reading an error message makes space available in the register for the next error.

ERR? May Clear the Status Register Error Bit

The status register error bit (bit 5) and the front panel ERR annunciator are cleared when the error register is empty.

RESET and CLR Clear the Error Register

The **RESET** (or **RST**) and **CLR** commands clear the error register, the status register error bit (bit 5), and the ERR annunciator.

Data Destination

If the **MEM** command is used, the value returned by **ERR?** is stored in the specified variable or array. If the **MEM** command is not used, the response is sent to the output buffer in the default ASCII format.

HP-IB Data Format

The HP 3245A returns numeric results in either ASCII or binary format (see **OFORMAT** command). In the default ASCII format, **ERR?** returns REAL results in 8-digit scientific notation. In the binary format, **ERR?** returns REAL results in IEEE-754 64-bit notation.

Related Commands

CLR, ERRSTR?, RESET, RST, STB?

Example

Example: Reading Error Codes

This program uses the **ERR?** command within a loop to read and print all error codes in the error register.

```
10  REPEAT                      !Beginning of loop
20    OUTPUT 709;"ERR?"         !Read error code
30    ENTER 709;Code            !Enter code (error number)
40    PRINT Code                !Print code
50  UNTIL Code=0                !Loop until all errors are read
60  END
```

A typical return for three errors (1, 62, and 2) follows. Note that the order of error occurrence in 1, 62, and 2.

```
1
62
2
0
```

ERRSTR?

Description

Error String Query. Returns the error code and string of the most recent error, deletes that error from the error register, and clears the status register error bit after all errors are read.

Syntax

ERRSTR?

Parameters

None.

Remarks

Data Returned

ERRSTR? returns a number, a comma, and a quoted string which may contain up to 256 characters. Errors are returned in a first-in-first-out fashion (i.e., the first error to occur is the first error to be read). If no error codes are in the error register, `" 0, NO ERROR"` is returned.

ERRSTR? Clears the Error Register

ERRSTR? reads the error messages and clears the error register one error at a time. Errors that occur while the register is full are displayed as usual, but are not stored in the error register. Reading an error message makes space available in the register for the next error.

ERRSTR? May Clear the Status Register Error Bit

The status register error bit (bit 5) and the front panel ERR annunciator are cleared when all errors have been read from the error register.

RESET and CLR Clear the Error Register

The **RESET** (or **RST**) or **CLR** commands clear the error register, the status register error bit, and the ERR annunciator.

Data Destination

Executing **ERRSTR?** from the front panel displays the response on the front panel display. When **ERRSTR?** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

Related Commands

CLR, ERR?, RESET, RST, STB?

Example

Example: Reading Error Messages

This program uses the **ERRSTR?** command within a loop to read and print all error messages in the error register.

```
10  DIM Message$[256]            !Dimension array
20  REPEAT                       !Beginning of loop
30    OUTPUT 709;"ERRSTR?"       !Read error register
40    ENTER 709;Code,Message$    !Enter code, message
50    PRINT Code,Message$        !Display code, message
60  UNTIL Code=0                 !Loop until all errors are read
70  END
```

A typical return for two errors (61 and 1 in that order) follows.

```
    61,  "OUT OF RANGE -- 100"
     1,  "INCOMPLETE COMMAND -- A; Expected command header or assignment"
     0,  "NO ERROR"
```

EXOR

Description

Exclusive-OR. Returns a "0" or "1" based upon the logical exclusive-OR of the specified arguments.

Syntax

argument **EXOR** *argument*

Parameters

argument Each *argument* parameter may be a number, numeric variable, math function, array element, or numeric expression (enclosed in parentheses).

Remarks

EXOR Operation

The **EXOR** command returns "0" or "1" as shown in the following truth table. If one argument specified is a non-zero value, "1" is returned. If both arguments are 0, or both are non-zero, a "0" is returned.

A	B	A EXOR B
0	0	0
0	1	1
1	0	1
1	1	0

Related Commands

BINEOR, OR

Example

Example: EXOR Statement in IF...END IF Loop (EXOR)

This program uses **EXOR** within an **IF...END IF** loop. The first time the subroutine is called, A = 3 and B = 0, so the A EXOR B result (line 90) is "TRUE". The second time the subroutine is called, A = 0, so the A EXOR B result is "FALSE".

```
10   !file EXOR
20   !
30   CLEAR 709                    !Clear 3245A
40   OUTPUT 709;"RST"             !Reset 3245A
50   OUTPUT 709;"SCRATCH"         !Clear 3245A memory
60   !
70   OUTPUT 709;"SUB EXORLOOP"    !Download subroutine
80   OUTPUT 709;"  INTEGER A,B"   !Define integers A and B
90   OUTPUT 709;"  IF A EXOR B THEN" !Start loop
100  OUTPUT 709;"    DISP 'TRUE'"  !Display "TRUE" if A EXOR B = 1
110  OUTPUT 709;"  ELSE"
120  OUTPUT 709;"    DISP 'FALSE'" !Display "FALSE" if A EXOR B = 0
130  OUTPUT 709;"  END IF"        !End loop
140  OUTPUT 709;"SUBEND"          !End subroutine
150  !
160  OUTPUT 709;"A=3;B=0"         !Define A and B
170  OUTPUT 709;"CALL EXORLOOP"   !Call subroutine
180  WAIT 2
```

```
190  OUTPUT 709;"A=0"               !Redefine A
200  OUTPUT 709;"CALL EXORLOOP"     !Call subroutine
210  END
```

EXP

Description

Exponent. Raises the base e (2.718281828) to the power of the argument.

Syntax

EXP (*argument*)

Parameters

argument — The *argument* must be a number or numeric expression (enclosed in parentheses).

Remarks

Related Commands

LGT, LOG

Example

Example: Using the EXP Function

This program raises e to the 10th power and displays the result (22026.466) on the controller CRT.

```
10  OUTPUT 709;"VREAD EXP(10)"  !Raise e to 10th power, read value
20  ENTER 709;A                 !Enter result
30  PRINT "Value = ";A          !Display result
40  END
```

A typical return is:

```
    Value = 22026.466
```

Description

Fast Amplitude/Offset Change. Allows fast change of AC waveform amplitude and/or DC offset (less than 90 µsec from a subroutine).

Syntax

FASTAMP ***ch, amplitude, offset*** (sine/ramp/arb waveforms)

FASTAMP ***ch, high_level, low_level*** (square waveforms)

Parameters

ch **USE** Channel. 0 sets channel A, 100 sets channel B.

amplitude For sine, ramp, and arbitrary waveforms, *amplitude* is an integer from 0 (nominally 0 amplitude) to 1777 (amplitude nominally equal to 100% of range).

offset For sine, ramp, and arbitrary waveforms, *offset* is an integer from -1319 (nominally equal to -50% of peak-to-peak range) to +1319 (nominally equal to +50% of peak-to-peak range).

high_level For square waveforms, *high_level* is an integer from -1682 (nominally equal to -50% of peak-to-peak range) to +1682 (nominally equal to +50% of peak-to-peak range).

low_level For square waveforms, *low_level* is an integer from +1612 (nominally equal to -50% of peak-to-peak range) to -1612 (nominally equal to +50% of peak-to-peak range).

Remarks

Using FASTAMP Command

Using **FASTAMP** allows amplitude and DC offset changes to be made faster than with **APPLY** or **DCOFF**, however; features such as queries (FREQ?), display monitoring (**MON STATE**), autoranging, parameter value checking and rounding, and automatic use channel either return invalid values, or cannot be used at all. **APPLY ACV/ACI**, **APPLY RPV/RPI**, **APPLY SQV/SQI**, or **APPLY WFV/WFI** must be executed before **FASTAMP** is executed for the first time. When finished using **FASTAMP**, **RESET** is recommended to correct queries (**FREQ?**), display monitoring (**MON STATE**), autoranging, etc.

Related Commands

APPLY, DCOFF, TRIGFREQ

Example

Example: Fast Amplitude/Offset Sweep (FASTAMP)

This program sweeps amplitude, DC offset, and frequency for both channel A and channel B. Note that using **TRIGFREQ** maintains precise phase relationship between the two channel outputs since the frequencies on both channels are changed simultaneously when **TRIGIN** is executed. If **FREQ** or **FASTFREQ** were used, some phase shift would occur sine the frequencies would not be simultaneously updated. This program requires about 860 µsec per iteration.

```
10    !file FASTAMP
20    !
30    ASSIGN @Source TO 709
40    N_points=3000                                  !Number sweep points
50    OUTPUT @Source;"RST"                           !Reset HP 3245A
60    OUTPUT @Source;"SCRATCH"                       !Clear HP 3245A memory
70    !
80    !Define arrays, variables
90    !
100   OUTPUT @Source;"DIM F0(";N_points;")"          !Freq array
110   OUTPUT @Source;"INTEGER AMP0(";N_points;")"    !Ch A amplitude array
120   OUTPUT @Source;"INTEGER AMP1(";N_points;")"    !Ch B amplitude array
130   OUTPUT @Source;"INTEGER OFF0(";N_points;")"    !Ch A offset array
140   OUTPUT @Source;"INTEGER OFF1(";N_points;")"    !Ch B offset array
150   OUTPUT @Source;"INTEGER I,A"                   !Def INTEGER vars
160   !
170   !Define ch A and B amplitude and DC offset arrays
180   !
190   OUTPUT @Source;"SUB DEF_ARRAYS"                !Define arrays
200   OUTPUT @Source;"  FOR I=1 TO ";N_points
210   OUTPUT @Source;"    F0(I)=I*100."
220   OUTPUT @Source;"    AMP0(I)=888+888.*I/";N_points  !Ch A ampl
230   OUTPUT @Source;"    AMP1(I)=888"               !Ch B ampl
240   OUTPUT @Source;"    OFF0(I)=0"                 !Ch A DC offset
250   OUTPUT @Source;"    OFF1(I)=660.*I/";N_points !Ch B DC offset
260   OUTPUT @Source;"  NEXT I"
270   OUTPUT @Source;"SUBEND"                        !End subroutine
280   OUTPUT @Source;"CALL DEF_ARRAYS"               !Call subroutine
290   !
300   !Set channel A triggering/output
310   !
320   OUTPUT @Source;"USE 0"                         !Use channel A
330   OUTPUT @Source;"TRIGIN HIGH"                   !Set TRIGIN HIGH
340   OUTPUT @Source;"TRIGOUT EXT"                   !Enable Trigger conn
350   OUTPUT @Source;"FREQ 0"                        !Hold at zero phase
360   OUTPUT @Source;"TRIGMODE ARMWF"                !Set armed mode
370   OUTPUT @Source;"APPLY ACV 1"                   !Apply sine wave
380   !
390   !Set channel B triggering/output
400   !
410   OUTPUT @Source;"USE 100"                       !Use ch A
420   OUTPUT @Source;"TRIGIN EXT"                    !External trigger
430   OUTPUT @Source;"FREQ 0"                        !Hold at zero phase
440   OUTPUT @Source;"TRIGMODE ARMWF"                !Set armed mode
450   OUTPUT @Source;"APPLY RPV 1"                   !Apply ramp wave
460   !
470   !Set initial values on channel A and B
480   !
490   OUTPUT @Source;"TRIGFREQ   0,F0(1)"            !ch A init freq
500   OUTPUT @Source;"TRIGFREQ 100,F0(1)"            !ch B init freq
510   OUTPUT @Source;"FASTAMP    0,AMP0(1),0"        !ch A ampl/offset
520   OUTPUT @Source;"FASTAMP  100,AMP1(1),0"        !ch B ampl/offset
530   !
```

```
540  !Change ch A and B amplitude, DC offset, frequency
550  !
560  OUTPUT @Source;"USE 0"                         !Use channel A
570  OUTPUT @Source;"SUB SWEEPA"                    !Define freq array
580  OUTPUT @Source;"  FOR I=1 TO ";N_points        !Sweep thru points
590  OUTPUT @Source;"    FASTAMP    0,AMP0(I),0"    !Update ch A ampl/off
600  OUTPUT @Source;"    FASTAMP  100,AMP1(I),0"    !Update ch B ampl/off
610  OUTPUT @Source;"    TRIGFREQ   0,F0(I)"        !ch A trig freq mode
620  OUTPUT @Source;"    TRIGFREQ 100,F0(I)"        !ch B trig freq mode
630  OUTPUT @Source;"    TRIGIN SGL"                !Change frequencies
640  OUTPUT @Source;"  NEXT I"
650  OUTPUT @Source;"SUBEND"
660  !
670  !Measure/display iteration time
680  !
690  T1=TIMEDATE
700  OUTPUT @Source;"SWEEPA"
710  T2=TIMEDATE
720  DISP (T2-T1)/N_points                          !Display iteration time
730  STOP
740  END
```

A typical return is:

```
.00086333211263
```

FASTFREQ

Description

Fast Frequency Change. Allows fast change of AC waveform frequency (less than 150 µsec from a subroutine).

Syntax

FASTFREQ *ch, freq_mHz*

Parameters

ch **USE** Channel. 0 sets channel A, 100 sets channel B.

freq_mHz Frequency in millihertz (fractional part will be truncated).

Remarks

Using FASTFREQ Command

Using **FASTFREQ** allows frequency changes to be made faster than with **FREQ**, however: features such as queries (e.g. FREQ?), display monitoring (**MON STATE**), autoranging, parameter value checking and rounding, and automatic use channel either return invalid values, or cannot be used at all. **APPLY ACV**, **APPLY RPV**, **APPLY SQV**, or **APPLY WFV** must be executed before **FASTFREQ** is executed the first time. When you are finished using **FASTFREQ**, **RESET** is recommended to correct queries (**FREQ?**), display monitoring (**MON STATE**), autoranging, etc.

Related Commands

FREQ, TRIGFREQ

Example

Example: Increasing Sweep Speed (FASTFREQ)

This program increases the sweep speed of a 1 Vac PP sine waveform output from channel A to 300 µsec per iteration. Note that the frequency is specified in millihertz. The frequency sweep is from 10 to 100 Hz in 0.1 Hz steps.

```
10   !file FASTFREQ
20   !
30   CLEAR 709                       !Clear HP 3245A
40   OUTPUT 709;"RST"                !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"            !Clear HP 3245A memory
60   OUTPUT 709;"REAL F"             !Define REAL variable
70   OUTPUT 709;"SUB SWEEP"          !Begin subroutine
80   OUTPUT 709;"  FOR F=10000 TO 100000 STEP 100"  !Sweep freq
90   OUTPUT 709;"    FASTFREQ 0,F"   !Fast freq sweep
100  OUTPUT 709;"  NEXT F"           !Increment count
110  OUTPUT 709;"SUBEND"             !End subroutine
120  OUTPUT 709;"CALL SWEEP"         !Call subroutine
130  OUTPUT 709;"USE 0"              !Use channel A
140  OUTPUT 709;"APPLY ACV 1"        !Apply 1 Vac PP sine wave
150  END
```

Description

Fetch Value. Returns the value of the specified variable, expression, or displayed quoted string. **FETCH** cannot be executed from the front panel.

Syntax

FETCH *variable* / *expression* / *string*

Parameters

variable

Variable name. Variable names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore character ("_"), or the question mark ("?"). Variable names must not be the same as HP 3245A commands or parameters or stored state names.

expression

Any valid number or numeric expression (enclosed in parentheses) may be used.

string

A quoted ASCII string which must be enclosed in double quotation marks (" "*string*" ") or apostrophes ('*string*').

Remarks

Data Returned

FETCH returns the value of the specified numeric variables or numeric expressions. If a quoted string is specified, **FETCH** displays the quoted string.

Data Destination

When **FETCH** is executed, the response is sent to the output buffer in the default ASCII format.

Related Commands

DISP, DSP, VREAD

Examples

Example: Fetching a Variable

This program assigns a value to variable A (3E2) and displays the value (300) on the controller CRT.

```
10  OUTPUT 709;"LET A=3E2"     !Assign value to A
20  OUTPUT 709;"FETCH A"       !Fetch value
30  ENTER 709;Var              !Enter result
40  PRINT Var                  !Display "300"
50  END
```

Example: Fetching a Quoted String

This program displays "HP 3245A" on the controller CRT.

```
10  OUTPUT 709;"FETCH 'HP 3245A'"   !Fetch string
20  ENTER 709;A$                    !Enter string
30  PRINT A$                        !Display "HP 3245A"
40  END
```

Example: Fetching a Quoted String and Variable

This program uses **FETCH** to display a message followed by the value of a variable containing the value of the cosine of .5235 radians.

```
10  DIM A$[50]                           !Dimension array
20  OUTPUT 709;"VAL = COS(.5235)"        !Compute cosine
30  OUTPUT 709;"FETCH 'Cosine = ',VAL"   !Fetch cosine value
40  ENTER 709;A$                         !Enter value
50  PRINT A$                             !Display value
60  END
```

A typical return is:

```
Cosine = 8.6607479E-01
```

Description

Fill Array. Places specified values into a previously dimensioned array (see the **DIM, INTEGER, REAL** commands). **FILL** is useful in defining array elements for outputs using **APPLY DCMEMI,APPLY DCMEMV, APPLY WFI, and APPLY WFV** commands.

Syntax

FILL *array_name, list*

Parameters

array_name

Array name. Array names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore character ("_"), or the question mark ("?"). Array names must not be the same as HP 3245A commands or parameters, or stored state names.

list

The *list* parameter specifies a list of numbers to be entered into the array. If used in a subroutine, the number of *list* items cannot exceed the subroutine size limit.

Remarks

User-Defined Arrays are Global

All arrays are global among all front panel, HP-IB, and subroutine operations. A user-defined array element may be used in any HP 3245A command where a numeric parameter is required.

Using FILL with APPLY WFV and APPLY WFI

If FILL loads an array to be used by APPLY WFV or APPLY WFI, the numbers entered must have values between ± 1 inclusive.

Related Commands

DIM, INTEGER, REAL

Examples

Example: Triggered-Sequence DC Voltage Outputs (FILL)

This program outputs a sine wave from channel A whose amplitude increases from 2V to 10V in 2V increments. The FILL command in line 70 loads an array with the desired amplitudes. A subroutine containing a FOR...NEXT loop is downloaded to the 3245A. Within the loop is the APPLY ACV command. The amplitudes for the command are contained in the array (VOUT) loaded by FILL. As each pass through the FOR...NEXT loop is made, the next amplitude in the array is specified (VOUT(I)) and is held for for two seconds. After the loop completes and 10V is applied, the output is set to 0V.

```
10   !file FILL
20   !
30   CLEAR 709                        !Clear 3245A
40   OUTPUT 709;"RST"                 !Reset 3245A
50   OUTPUT 709;"SCRATCH"             !Clear 3245A memory
60   OUTPUT 709;"REAL VOUT(4)"        !Dimension array
70   OUTPUT 709;"FILL VOUT 2,4,6,8,10" !Enter array values
```

```
80   !
90   OUTPUT 709;"SUB APPLY_AMP"        !Download subroutine
100  OUTPUT 709;"  USE 0"              !Specify USE channel (A)
110  OUTPUT 709;"  FOR I=0 TO 4"       !Begin FOR/NEXT loop
120  OUTPUT 709;"    APPLY ACV VOUT(I)" !Output = 2V, 4V, 6V, ...
130  OUTPUT 709;"    WAIT 2"           !Hold amplitude 2 sec
140  OUTPUT 709;"  NEXT I"
150  OUTPUT 709;"  APPLY DCV 0"        !Output = 0V
160  OUTPUT 709;"SUBEND"               !End subrouting
170  OUTPUT 709;"CALL APPLY_AMP"       !Execute subroutine
180  END
```

Description

Fill Array With Sine Waveform Values. Performs a scaling function on AC sine waveforms to decrease the time required for output.

Syntax

FILLAC *integer_array*

Parameters

integer_array

Array name. Name of the array containing 2048 precomputed INTEGER values. Array names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore character ("_"), or the question mark ("?"). Array names must not be the same as HP 3245A commands or parameters, or stored state names.

Remarks

Precomputing (Scaling) Operation

Whenever an **APPLY** command is used to generate a sine waveform, the HP 3245A downloads 2048 points to the **USE** channel. To arithmetically scale the data points into a format which the channel can use requires approximately 20 msec for sine waveforms. In addition to the scaling time, approximately 70 msec are required to download the data points.

To reduce the time required to generate the output, **FILLAC** can be used to perform a scaling function on the sine waveform and store the precomputed values into an INTEGER array. The values stored in the INTEGER array are then output from the channel using the **APPLY WFV** command. Thus, with precomputation, the sine wave output is generated about 70 msec after **APPLY WFV** is executed, as compared to 115 msec without precomputation.

Related Commands

APPLY WFV, FILLRP, FILLWF, USE

Example

Example: Precomputing AC Sine Waveform (FILLAC)

This program precomputes and outputs a 10 kHz sine waveform with an amplitude of 5 Vac (PP) from channel A. Since autorange is enabled (power-on state), the channel outputs the waveform on the 5 VPP range. The **FILLAC** command (line 90) precomputes 2048 points to define a normalized (1 Vac PP, 1 kHz) sine wave.

Next, **FREQ** (line 100) sets the output frequency to 10 kHz. Then **APPLY WFV** (line 110) uses the normalized point values values along with the frequency setting of 10 kHz and amplitude of 5 Vac (PP) to generate a 5 Vac (PP), 10 kHz sine waveform.

```
10   !file FILLAC
20   !
30   CLEAR 709                         !Clear 3245A
40   OUTPUT 709;"RST"                  !Reset 3245A
50   OUTPUT 709;"SCRATCH"              !Clear 3245A memory
60   !
70   OUTPUT 709;"INTEGER SCALE(2047)"  !Declare INTEGER array
```

```
80   OUTPUT 709;"USE 0"                    !Use channel A
90   OUTPUT 709;"  FILLAC SCALE"           !Fill array with scaled values
100  OUTPUT 709;"  FREQ 10E3"              !Set frequency to 10 kHz
110  OUTPUT 709;"  APPLY WFV 5,SCALE"      !Output 10 kHz sine wave
120  END
```

Description

Fill Array With Binary Values. Places specified binary values into a previously dimensioned array (see **DIM, REAL, INTEGER** commands).

Syntax

FILLBIN *array_name, block_data*

Parameters

array_name Array name. Array names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore character ("_"), or the question mark ("?"). Array names must not be the same as HP 3245A commands or parameters, or stored state names.

block_data The *block_data* parameter consists of the # sign, the letter A, and a 16-bit integer which indicates the number of bytes of data to follow.

Remarks

Related Commands

DIM, FILL, INTEGER, REAL

Example

Example: Fill Array With Binary Values (FILLBIN)

This program transfers binary data (a Gaussian noise pattern) from a Series 200/300 controller to the 3245A. The data is preceded by the IEEE 728 Block A header which consists of #, A, and a 16-bit integer (Lgnth) indicating the number of bytes which follow the header. The signal is a 2048 point waveform with eight bytes per point/reading.

```
10   !file FILLBIN
20   !
30   CLEAR 709                         !Clear 3245A
40   OUTPUT 709;"RST"                  !Reset 3245A
50   OUTPUT 709;"SCRATCH"              !Clear 3245A memory
60   OUTPUT 709;"REAL WV_AMP(2047)"    !Array to be filled by controller
70   !
80   ASSIGNBin_dat TO 709;FORMAT OFF  !Assign I/O path to HP 3245A
90   REAL Wv_amp(0:2047)               !Controller array w/Gaussian Noise
100  INTEGER Lngth                     !Variable w/# bytes after header
110  Lngth=16384                       !# of bytes which follow header
120  FOR I=0 TO 2047                   !Generate Gaussian noise pattern
130  Wv_amp(I)=.4*(RND+RND+RND+RND+RND)-1
140  NEXT I
150  !
160  OUTPUT 709;"USE 0"                !Use channel A
170  OUTPUT 709;"  FILLBIN WV_AMP,#A";  !Transfer data to 3245A
180  OUTPUTBin_dat;Lngth;wV_amp(*)     !with header preceding data
190  OUTPUT 709;"  APPLY WFV 2.5,WV_AMP" !Apply dat a as an a  rb waveform
200  END
```

FILLRP

Description

Fill Array With Ramp Waveform Values. Performs a scaling function on ramp waveforms to decrease the time required for output.

Syntax

FILLRP *integer_array* [, *%_duty*]

Parameters

integer_array

Name of the array containing 2048 precomputed INTEGER values. Array names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore character ("_"), or the question mark ("?"). Array names must not be the same as HP 3245A commands or parameters, or stored state names.

%_duty

Duty cycle between 5% and 95%. Default *%_duty* = 50%.

Remarks

Precomputing (Scaling) Operation

Whenever an **APPLY** command is used to generate a ramp waveform, the HP 3245A downloads 2048 points to the **USE** channel. To arithmetically scale the data points into a format which the channel can use requires approximately 45 msec for ramp waveforms. In addition to the scaling time, approximately 70 msec are required to download the data points.

To reduce the time required to generate the output, **FILLRP** can be used to perform a scaling function on the ramp waveform and store the precomputed values into an INTEGER array. The values stored in the INTEGER array are then output from the channel using the **APPLY WFV** command. Thus, with precomputation, the ramp wave output is generated about 70 msec after **APPLY WFV** is executed, as compared to 115 msec without precomputation.

Related Commands

APPLY WFV, FILLAC, FILLWF, USE

Example

Example: Precomputing AC Ramp Waveform (FILLRP)

This program precomputes and outputs a 3 kHz ramp waveform with an amplitude of 3.25 VPP from channel A. The waveform duty cycle is set to 33%. Since autorange is enabled (power-on state), the channel outputs the waveform on the 5 VPP range.

```
10   !file FILLRP
20   !
30   CLEAR 709                      !Clear 3245A
40   OUTPUT 709;"RST"               !Reset 3245A
50   OUTPUT 709;"SCRATCH"           !Clear 3245A memory
60   !
70   OUTPUT 709;"INTEGER J(2047)"   !Define INTEGER array J
80   OUTPUT 709;"USE 0"             !Use channel A
90   OUTPUT 709;"  FILLRP J,33"     !Fill J, duty cycle = 33%
100  OUTPUT 709;"  FREQ 3E3"        !Change output freq to 3 kHz
110  OUTPUT 709;"  APPLY WFV 3.25,J" !Output ramp waveform
120  END
```

Description

Fill Array With Arbitrary Waveform Values. Performs a precomputing function on arbitrary waveforms to decrease the time required for output.

Syntax

FILLWF *real_array, integer_array*

Parameters

real_array

Name of the REAL array containing up to 2048 arbitrary waveform points which will be converted to integer format (by **FILLWF**) and stored in *integer_array*. Array names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore character ("_"), or the question mark ("?"). Array names must not be the same as HP 3245A commands or parameters, or stored state names.

integer_array

Name of the INTEGER array containing 2048 precomputed integer values. Array names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore character ("_"), or the question mark ("?"). Array names must not be the same as HP 3245A commands or parameters, or stored state names.

Remarks

Precomputing (Scaling) Operation

Whenever an **APPLY** command is used to generate an arbitrary waveform, the HP 3245A downloads 2048 points to the **USE** channel. To arithmetically scale the data points into a format which the channel can use requires approximately 500 msec for arbitrary waveforms. In addition to the scaling time, approximately 70 msec are required to download the data points.

To reduce the time required to generate the output, **FILLWF** can be used to perform a scaling function on the arbitrary waveform and store the precomputed values into an INTEGER array. The values stored in the INTEGER array are then output from the channel using the **APPLY WFV** command. Thus, with precomputation, the arbitrary waveform output is generated about 70 msec after **APPLY WFV** is executed, as compared to 570 msec without precomputation.

Related Commands

APPLY WFV, FILLAC, FILLRP, USE

Example

Example: Precomputing an Arbitrary Waveform (PRE_ARB)

This program demonstrates the speed advantage of precomputing arbitrary waveforms. In the example, the arbitrary waveform Sin(x)/x is computed and stored in a 3245A real array. The array is then specified by the APPLY WFV command and command execution is timed. FILLWF precomputes Sin(x)/x using the data in the real array and stores the results in an integer array. The (integer) array is then specified by APPLY WFV whose execution is timed again. The times shown following the program listing show the advantage of using precomputed data.

```
10   !file PRE_ARB
20   !
30   REAL A(0:1)                        !Controller variable for exec. times
40   CLEAR 709                          !Clear 3245A
50   OUTPUT 709;"RST"                   !Reset 3245A
60   OUTPUT 709;"SCRATCH"               !Clear 3245A memory
70   !
80   OUTPUT 709;"OUTBUF ON"             !Enable output buffer
90   OUTPUT 709;"INTEGER SCALE(2047)"   !Array for precomput  ed data
100  OUTPUT 709;"REAL T(3)"             !Array for execution times
110  !
120  OUTPUT 709;"SUB SINX"              !Subroutine to compute Sin(x)/x
130  OUTPUT 709;"  REAL WV_AMP(2047)"   !REAL array for Sin(x)/x data
140  OUTPUT 709;"  PI=3.1415"           !Assign value to PI
150  OUTPUT 709;"  X=(-5*PI+10*PI/2048)" !Initialize X a
160  OUTPUT 709;"  FOR I = 0 TO 2047"   !!Evaluate Sin(x)/x
170  OUTPUT 709;"    IF X = 0.0 THEN"   !Test for X = 0
180  OUTPUT 709;"      WV_AMP(I) = 1.0"
190  OUTPUT 709;"    ELSE"
200  OUTPUT 709;"      WV_AMP(I) = SIN(X)/X"
210  OUTPUT 709;"    END IF"
220  OUTPUT 709;"    X=X+(10*PI/2048)"
230  OUTPUT 709;"  NEXT I"
240  OUTPUT 709;"SUBEND"
250  OUTPUT 709;"CALL SINX"
260  !
270  OUTPUT 709;"USE 0"                  !Use channel A
280  OUTPUT 709;"MEM T"                  !Enable 3245A memory
290  OUTPUT 709;"TIME"                   !Read 3245A clock
300  OUTPUT 709;"  APPLY WFV 2,WV_AMP"   !Command which is timed
310  OUTPUT 709;"TIME"                   !Read 3245A clock
320  !
330  OUTPUT 709;"FILLWF WV_AMP, SCALE"   !Scale Sin(x)/x, store in array
340  OUTPUT 709;"TIME"                   !Read 3245A clock
350  OUTPUT 709;"  APPLY WFV 2,SCALE"    !Time precomputed waveform
360  OUTPUT 709;"TIM  E"                   !Read 3245A clock
370  OUTPUT 709;"MEM OFF"                !Disable memory
380  !
390  OUTPUT 709;"VREAD T(1)-T(0)"        !Compute execution time
400  OUTPUT 709;"VREAD T(3)-T(2)"        !Compute execution time (precomp)
410  ENTER 709;A(*)                      !Enter computed times
420  PRINT "ARBITRARY WAVEFORM =";A(0)
430  PRINT
440  PRINT "PRECOMPUTED ARBITRARY WAVEFORM =";A(1)
450  END
```

A typical output based on the program is shown below:

```
ARBITRARY WAVEFORM = .73

PRECOMPUTED ARBITRARY WAVEFORM = .11
```

Description

For-Next Loop. Defines a loop which is repeated until a loop counter passes a specified value. For-Next loops can only be used in 3245A subroutines.

Syntax

FOR *counter* = *initial_value* **TO** *final_value* **[STEP** *step_size***]**

program segment

NEXT *counter*

Parameters

counter The *counter* parameter is a variable name which acts as the loop counter.

initial_value The *initial_value* parameter is a number, numeric variable, or numeric expression (enclosed in parentheses) which is the beginning value of the loop counter. The loop counter is set to this value when the loop is entered.

final_value The *final_value* parameter is a number, numeric variable, or numeric expression (enclosed in parentheses) which is the ending value of the loop counter. When the loop counter exceeds this value, program execution continues with the statement after the NEXT statement.

step_size The optional *step_size* parameter is a number or numeric expression (enclosed in parentheses) which specifies the amount the loop counter is incremented or decremented for each pass through the loop. A negative *step_size* decrements the loop counter for each pass. Default *step_size* = 1.

Remarks

FOR...NEXT is Valid Only in Subroutines

The **FOR...NEXT** construct may be used only within subroutines.

Using the FOR...NEXT Construct

The loop counter is set equal to *initial_value* when the loop is first entered. Each time the NEXT statement is encountered, the *counter* increments by *step_size* and is tested against *final_value*. If *final_value* is not exceeded, program execution continues with the line immediately following the FOR statement.

When *counter* exceeds *final_value*, program execution continues with the line immediately following the NEXT statement. Notice that the loop counter is one greater than the final value when the loop is exited (or one less than if step size is negative).

The initial value, final value, and step size are calculated when the loop is entered. These values are used while the loop is repeating. If a variable or expression is used for any of these values, its value may be changed after entering the loop without affecting the number of times the loop is repeated. However, changing the value of the loop counter affects the number of times the loop is repeated.

Related Commands

IF. . .THEN, SUB, SUBEND, WHILE

Examples

Example: FOR...NEXT Loop

This program sets up a **FOR...NEXT** loop with an initial value of 1 and a final value of 20 (default step = 1). The loop is part of subroutine FORLOOP which displays the value of the counter (1 through 20) when the subroutine is executed.

```
10  OUTPUT 709;"INTEGER I"           !Define INTEGER variable I
20  OUTPUT 709;"SUB FORLOOP"         !Begin subroutine
30  OUTPUT 709;"  FOR I=1 TO 20"     !Begin loop
40  OUTPUT 709;"    DISP I;WAIT .5"  !Display value; wait .5 sec.
50  OUTPUT 709;"  NEXT I"            !Next value
60  OUTPUT 709;"SUBEND"              !End subroutine
70  OUTPUT 709;"CALL FORLOOP"        !Execute subroutine
80  END
```

Example: FOR...NEXT Loop with Negative Step

This program sets up a **FOR...NEXT** loop with an initial value of 20, a final value of 1, and a step size of -1. The loop is part of subroutine LOOP which displays the value of the counter (20 through 1) when the subroutine is executed.

```
10  OUTPUT 709;"INTEGER I"                   !Define INTEGER variable I
20  OUTPUT 709;"SUB LOOP"                    !Begin subroutine
30  OUTPUT 709;"  FOR I=20 TO 1 STEP -1"     !Begin loop
40  OUTPUT 709;"    DISP I;WAIT .5"          !Display value; wait .5 sec.
50  OUTPUT 709;"  NEXT I"                    !Next value
60  OUTPUT 709;"SUBEND"                      !End subroutine
70  OUTPUT 709;"CALL LOOP"                   !Execute subroutine
80  END
```

Description

Set/Read Output Frequency. **FREQ** sets the output frequency for sine, ramp, square, and arbitrary waveforms on the **USE** channel. **FREQ** also sets the high-trigger-level and low-trigger-level frequencies when using the dual-frequency mode. **FREQ?** returns the high-trigger-level and low-trigger-level frequency settings on the **USE** channel.

Syntax

FREQ *freq_1* [, *freq_2*]

FREQ?

Parameters

freq_1

This parameter specifies the single output frequency for normal operation or the high-trigger-level output frequency in dual-frequency mode. The output frequency range for sine, square, and arbitrary waveforms is 0 to 1 MHz with 0.001 Hz resolution. Ramp waveform performance is not specified above 100 kHz and will degrade substantially above this frequency.

When using the dual-frequency mode (**TRIGMODE DUALFR**) the *freq_1* parameter specifies the output frequency generated when the input trigger is high (+5V). Power-on/reset *freq_1* = 1000 Hz.

freq_2

This optional parameter is used only in the dual-frequency mode and specifies the low-trigger-level output frequency. The value selected for this parameter is the output frequency generated when the input trigger is low (0V). The output frequency range for sine, square, and arbitrary waveforms is 0 to 1 MHz with 0.001 Hz resolution. Ramp waveform performance is not specified above 100 kHz and will degrade substantially above this frequency. Power-on/reset *freq_2* = 1000 Hz, while default *freq_2* = present *freq_2* value.

Remarks

Square Wave Duty Cycle Cannot Be Varied Above 100 kHz

At power-on, the duty cycle for ramp and square waveforms is set to 50% (see the **DUTY** command). The duty cycle can be varied from 5% to 95%. However, when generating square waveforms at frequencies above 100 kHz, an error is generated if the duty cycle is set to a value other than 50% (**DUTY 50**).

Query Command (FREQ?)

FREQ? returns the high-trigger-level and low-trigger-level frequency settings on the **USE** channel.

Related Commands

APPLYs, DUTY, TRIGIN, TRIGMODE, USE

Examples

Example: Setting the Output Frequency

The following program outputs a 5 Vp-p, 10 kHz sine wave.

```
10   CLEAR 709
20   OUTPUT 709;"RST"
30   OUTPUT 709;"SCRATCH"
40   !
50   OUTPUT 709;"USE 0"
60   OUTPUT 709;"  FREQ 10E3"
70   OUTPUT 709;"  APPLY ACV 5"
80   END
```

Example: The Dual Frequency Mode

The following program outputs a 5 Vp-p sine wave at 10 kHz or 100 kHz, depending on the input trigger level (TRIGIN).

```
10   CLEAR 709
20   OUTPUT 709;"RST"
30   OUTPUT 709;"SCRATCH"
40   !
50   OUTPUT 709;"USE 0"
60   OUTPUT 709;"  TRIGMODE DUALFR"
70   OUTPUT 709;"  TRIGIN HIGH"
80   OUTPUT 709;"  FREQ 10E3,100E3"
90   OUTPUT 709;"  APPLY ACV 5"
100  END
```

As the program executes, the 3245A is cleared, reset, and channel A is selected as the USE channel. TRIGMODE DUALFR (line 60) places the 3245A in the dual frequency mode. TRIGIN HIGH (line 70) sets the input trigger high so that when the signal is applied (line 90), the frequency is 10 kHz (line 80). Executing:

```
OUTPUT 709;"TRIGIN LOW"
```

sets the input trigger low, thus setting the output frequency to 100 kHz. Setting the input trigger high again would change the frequency to 10 kHz.

FTEST (FTST)

Description

Fixtured Self-Test. **FTEST** (or **FTST**) performs a fixtured self-test on the specified **USE** channel.

Syntax

FTEST [*ch*,[*ch*]]

Parameters

ch Channel parameter. *ch* = 0 (channel A), = 100 (channel B), = 1 (channel A rear panel), = 101 (channel B rear panel). If no channel is specified, channels A and B (front panel) are tested. Note that **FTEST** resets the specified channel(s).

Remarks

Data Returned

FTEST returns "PASS" if all tests pass or "FAIL" if one or more tests fail. Any failures that occur during the test are displayed on the front panel display and the first four errors are stored in the error register. (Use the **ERR?** or **ERRSTR?** command to read the error register.) Status register bit 5 (Error) is set if any failures occur.

Data Destination

When **FTEST** is executed from the front panel, the response is displayed on the front panel display. When **FTEST** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

Cabling Required for Fixtured Test

To perform the fixtured test, the Output port (front or rear panel) of the channel being tested must be connected to the channel's Trigger (I/O) port.

Related Commands

DTEST

Example

Example: Fixtured Self-Test on Channel A (FTEST)

This program performs a fixtured self-test on channel A. If all tests pass, "Test Passed" appears on the controller. If one or more tests fail, "Test Failed" appears on the controller and the first four errors are displayed. Note that the channel's Output port must be connected to its Trigger (I/O) port.

```
10   !file FTEST
20   !
30   DIM Err$[60]                 !Dimension controller array
40   OUTPUT 709;"USE 0"           !Use channel A
50   OUTPUT 709; "FTEST 0"        !Perform fixtured self-test
60   ENTER 709;A$                 !Enter result of test (pass/fail)
70   IF A$ = "FAIL" THEN          !Enter loop
80     PRINT "Test Failed"        !Display message if test fails
90     FOR I = 1 TO 4             !Error loop
100      OUTPUT 709;"ERRSTR?"     !Read error string
110      ENTER 709;Err$           !Enter string
```

```
120      PRINT Err$            !Print string
130    NEXT I                  !Loop until error register is empty
140  ELSE
150    PRINT "Test Passed"     !Display message if test passes
160  END IF                    !End loop
170  END
```

Should the 3245A fail a fixtured self test, return the instrument to the nearest Hewlett-Packard Sales and Service Office for repair.

Description

Help Function. Provides syntax statements and brief descriptions of selected HP 3245A commands and parameters.

Syntax

HELP [*topic*]

Parameters

topic

Type **HELP** *topic* for help on a specific command. For example, **HELP RESET** displays the command syntax and a brief description of the **RESET** command.

Remarks

Using HELP with "Two-Word" Commands

For help with commands such as APPLY DCV, APPLY ACV, etc., simply specify HELP APPLY. For help with the SET TIME command, specify HELP SET_TIME.

Examples

Example: Using HELP with SET TIME

This program uses the HELP function for the SET TIME command.

```
10  DIM A$[255]                !Dimension controller array
20  OUTPUT 709;"HELP SET_TIME" !Ask for help with SET TIME
30  ENTER 709;A$               !Enter information on SET TIME
40  PRINT A$                   !Print information on SET TIME
50  END
```

A typical return is:

```
SET_TIME: SET TIME <seconds> ; Sets the time of day clock to the specified
time.
```

Example: Using HELP with CALL

This program uses the HELP function for the CALL command.

```
10  DIM A$[255]               !Dimension controller array
20  OUTPUT 709;"HELP CALL"    !Help function for CALL command
30  ENTER 709;A$              !Enter string
40  PRINT A$                  !Display string
50  END
```

A typical return is:

```
    CALL: CALL <sub_name> ; Executes the named sub and waits for it
    to complete before executing any more commands.  EX: CALL Setup1;
```

ID?

Description **Model Number Query.** Returns the model number of the HP 3245A.

Syntax **ID?**

Parameters None.

Remarks

Data Returned

The ID? command returns "HP3245".

Related Commands

IDN?, REV?, SER?

Example

Example: Reading HP 3245A Model Number

This program uses ID? to read the model number of the HP 3245A.

```
10  OUTPUT 709;"ID?"        !Query HP 3245A model number
20  ENTER 709;A$            !Return HP 3245A model number
30  PRINT A$                !Display model number
40  END
```

A typical return is:

```
    HP3245
```

Description

Identity Query. Returns the manufacturer's name, the model number, the serial number, and the firmware revision number for an HP 3245.

Syntax

IDN?

Parameters

None.

Remarks

Data Returned

The **IDN?** command returns four lines of information. Line 1 is the manufacturer's name (HEWLETT PACKARD), line 2 is the instrument model number (3245), line 3 is the instrument serial number (always 0), and line 4 is the firmware revision number.

The firmware revision number is a four-digit year and date code. The code has the form "yyww", where "yy" is the year minus 1960, and "ww" is the week number of that year. For example, 2830 means that the latest firmware revision was the thirtieth week of 1988 (1988 - 1960 = 26).

Data Destination

When **IDN?** is executed from the front panel, the response is displayed on the front panel display. When **IDN?** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

Related Commands

ID?, REV?, SER?

Example

Example: Using the IDN? Command

This program demonstrates the use of the **IDN?** command.

```
10  OUTPUT 709;"IDN?"     !Read HP 3245A identity
20  FOR I = 1 TO 4        !Set loop counter
30    ENTER 709; A$       !Enter one string at a time
40    PRINT A$            !Print one string
50  NEXT I                !Increment counter
60  END
```

A typical return is:

```
HEWLETT PACKARD
3245
0
2833
```

IF...END IF

Description

If-End If Branching. Provides conditional branching within a subroutine.

Syntax

IF *expression* **THEN**

program segment

[**ELSE**]

[program segment]

END IF

Parameters

expression — A Boolean or numeric expression that is evaluated as TRUE if non-zero, or FALSE if evaluated as zero.

Remarks

IF...END IF is Valid Only in Subroutines

The **IF...END IF** construct may be used only within subroutines and the subroutine must be called (with the **CALL** command) or run (with the **RUN** command) to execute the commands.

Using the ELSE Statement

The **ELSE** statement is optional but, if used, must appear before the **END IF** statement. If *expression* is TRUE, execution continues with the program segment between **IF...THEN** and **ELSE**.

If *expression* is FALSE, execution continues with the segment after **ELSE**. In either case, when the program segment is completed, assuming there are no other loops or conditional branches, program execution continues with the first statement after the **END IF** statement.

Related Commands

FOR...NEXT, SUB, SUBEND, WHILE...END WHILE

Example

Example: Using the IF...END IF Function (IFENDIF)

This program uses **IF...END IF** within subroutine IFSQR to display the square root of a number or to indicate if the number is less than 0. The first time the subroutine is called, the value 2.0000000E+00 appears on the front panel display. The second time the subroutine is called, the number is less than 0, so the HP 3245A beeps once and "Number <0" appears on the front panel display.

```
10   !file IFENDIF
20   !
30   CLEAR 709                         !Clear 3245A
40   OUTPUT 709;"RST"                  !Reset 3245A
50   OUTPUT 709;"SCRATCH"              !Clear 3245A memory
60   !
70   OUTPUT 709;"REAL R"               !Define REAL variable
80   OUTPUT 709;"SUB IFSQR"            !Define subroutine IFSQR
90   OUTPUT 709;"  IF R<0 THEN"        !Start IF...END IF loop
100  OUTPUT 709;"    BEEP"             !BEEP once
110  OUTPUT 709;"    DISP 'Number <0'" !Display "Number <0" if R<0
120  OUTPUT 709;"  ELSE"               !Do 70-90 only if R>=0
130  OUTPUT 709;"    DISP SQR(R)"      !Display square root of R
140  OUTPUT 709;"  END IF"             !End IF...THEN loop
150  OUTPUT 709;"SUBEND"               !End subroutine
160  OUTPUT 709;"R=4"                  !Set R = 4
170  OUTPUT 709;"CALL IFSQR"           !Execute subroutine
180  OUTPUT 709;"WAIT 2"               !Wait 2 seconds
190  OUTPUT 709;"R=-1"                 !Set R = -1
200  OUTPUT 709;"CALL IFSQR"           !Execute subroutine
210  END
```

IMP/IMP?

Description

Set/Read Output Impedance. **IMP** selects 0Ω or 50Ω output impedance mode for DC and AC voltage outputs. **IMP?** returns the impedance setting on the **USE** channel.

Syntax

IMP *mode*

IMP?

Parameters

mode Output impedance mode. **IMP 0** selects the 0Ω mode, while **IMP 50** selects the 50Ω mode. Power-on/reset *mode* = 0.

Remarks

Using the IMP Command

In the power-on mode (**IMP 0**), the voltage output from the **USE** channel equals the programmed output voltage with or without load termination. For example, if you program channel A to output 2 VDC in the 0Ω mode, the actual output voltage will be 2 VDC with or without load termination.

However, in the 50Ω mode (**IMP 50**), the actual output voltage equals the programmed output voltage ONLY when the channel is terminated with a 50Ω load. For example, if you program channel A to output 2 VDC in the 50Ω mode, the actual output voltage will be 2 VDC only when channel A is terminated with a 50Ω load. If the load is removed and replaced with an open circuit, the actual output voltage will double to 4 VDC.

Changing Output Impedance Automatically Enables Autorange

When the output impedance is changed from 0Ω to 50Ω or vice versa, the autorange mode is automatically enabled. Refer to the **ARANGE** command for details on autoranging.

OUT OF RANGE Error

If the present DC or AC output voltage in the 0Ω mode is greater than the maximum output allowed in the 50Ω mode, selecting **IMP 50** will generate an "OUT OF RANGE" error. For example, if you set channel A to output 8 VDC in the 0Ω mode, selecting **IMP 50** will generate the error "OUT OF RANGE ---8.000000E+0". Note that the error value refers to the amplitude rather than to the impedance.

Similarly, if the present AC output voltage in the 50Ω mode is less than 0.03125 V ac PP (the smallest peak-to-peak voltage which can be generated in the 0Ω mode), selecting **IMP 0** will generate an "OUT OF RANGE" error.

Using the 50Ω Mode

Sine, ramp, square, and arbitrary waveforms output in the 0Ω (IMP 0) mode may not drive 50Ω cables properly at frequencies greater than 100 kHz. Therefore, the 50Ω (IMP 50) mode should be selected when using 50Ω cables at those frequencies. Generating high frequency waveforms with amplitudes greater than 10 V ac PP in the 0Ω mode may also cause the 3245A's current limited output to generate unpredictable waveforms.

Query Command (IMP?)

The **IMP?** command returns the impedance setting (0Ω or 50Ω) on the **USE** channel.

Related Commands

APPLYs, ARANGE, USE

Example

Example: Setting Output Impedance Mode (IMP)

The following program outputs a 200 kHz, 1 V ac PP square wave. Since the 50Ω mode is selected, the amplitude will be 1 V ac PP if a 50Ω load is at the output. If the load is not at the output, the amplitude will be 2 V ac PP.

```
10   !file IMP
20   !
30   CLEAR 709                  !Clear 3245A
40   OUTPUT 709;"RST"           !Reset 3245A
50   OUTPUT 709;"SCRATCH"       !Clear 3245A memory
60   !
70   OUTPUT 709;"USE 0"         !Use channel A
80   OUTPUT 709;"  FREQ 200E3"  !Set frequency to 200 kHz
90   OUTPUT 709;"  IMP 50"      !Set impedance to 50 ohms
100  OUTPUT 709;"  APPLY SQV 1" !Output a 1 V ac PP square wave
110  END
```

INBUF

Description

Enable/Disable Input Buffer. Enables or disables the HP 3245A input buffer. When enabled, the input buffer temporarily stores commands as it receives them over the HP-IB. This releases the bus immediately after each command is received, allowing the controller to perform other tasks while the HP 3245A is executing commands.

Syntax

INBUF *mode*

Parameters

mode The *mode* parameters follow. Power-on/reset *mode* = OFF.

mode	Definition
OFF	Disables input buffer - commands are accepted only when HP 3245A is not busy.
ON	Enables input buffer - commands are stored, thus releasing the bus immediately.

Remarks

Input Buffering OFF (INBUF OFF)

If input buffering is OFF and the HP 3245A is addressed to listen (i.e., accept commands from the controller), the HP 3245A accepts commands from the HP-IB one command at a time, as it is ready for them. While the instrument is processing the command, it holds the HP-IB.

After processing the command, the bus is released and the next command is accepted. Thus, the controller and the HP 3245A work in synchronization. HP-IB interface commands, such as Device Clear, Local, and Group Execute Trigger, act in sequence with other incoming commands.

Input Buffering ON

If input buffering is ON and the HP 3245A is addressed to listen (i.e., accept commands from the controller), the HP 3245A stores incoming commands in a 128-character input (circular) buffer. Commands are stored as fast as they are sent. This frees the controller for other activities while the HP 3245A processes the command string. If the input buffer fills, no more commands are accepted until space becomes available in the buffer.

When the input buffer is ON, synchronization is lost with the controller. However, you can monitor the ready bit in the HP 3245A Status Register (see the **READY?** command) to determine when the contents of the input buffer have been executed.

Related Commands

OUTBUF

Example

Example: Enabling the Input Buffer (INBUF)

The following example demonstrates how the 3245A's input buffer affects HP-IB programming.

As the program executes, the 3245A (and its memory) are cleared and reset. In line 70, the input buffer is enabled. In line 80, a sequence of commands is sent to the 3245A. Because the buffer is enabled, each of the commands sent are stored, thus releasing the bus and allowing the controller to display "Buffer Commands Executing". Once all commands in the buffer have executed, "Buffer Commands Complete" is displayed.

If the input buffer is disabled when the program executes, the 3245A holds the bus until each command in line 80 executes. Therefore, "Buffer Commands Executing" immediately followed by "Buffer Commands Complete" are not displayed until af   ter the commands in line 80 have finished.

```
10   !file INBUF
20   !
30   CLEAR 709                    !Clear 3245A
40   OUTPUT 709;"RST"             !Reset 3245A
50   OUTPUT 709;"SCRATCH"         !Clear 3245A memory
60   !
70   OUTPUT 709;"INBUF ON"        !Enable input buffer
80   OUTPUT 709;"USE 0;DELAY 1.5;APPLY DCV 1.0;APPLY DCV 2.0;READY?"
90   DISP "Buffer Commands Executing"
100  ENTER 709;A
110  IF A=1 THEN
120    DISP "Buffer Commands Complete"
130  END IF
140  END
```

INTEGER

Description

Dimension INTEGER Array or Variable. Reserves memory space in the HP 3245A to store integer variables or arrays.

Syntax

INTEGER *name* [(*max_index*)] [,*name* [(*max_index*)], ...]

Parameters

name

Variable or array name. Variable or array names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore character ("_"), or the question mark ("?"). Variable or array names must not be the same as HP 3245A commands or parameters, or stored state names.

The *name* parameter specifies the name of an INTEGER array or variable. Use *name* (*max_index*) to dimension an INTEGER array (e.g., `INTEGER TEST1(49)`). To dimension an INTEGER variable, omit the *max_index* parameter (e.g., `INTEGER VAR1`).

max_index

The *max_index* parameter specifies the number of elements in the array. The valid range is 0 to 32767. Since the lower bound (option base) for all arrays is zero, the number of elements in the array is one more than *max_index*. For example, "INTEGER A(5)" reserves memory space for array A with six elements (0 through 5).

Remarks

Redimensioning Arrays

Executing the **INTEGER** command defines an INTEGER array and fills all elements with zeroes. The redimensioned array is initialized to zeroes in all elements (if the array is redimensioned to the same size, it is cleared and all previous data is lost).

User-Defined Arrays are Global

All INTEGER arrays and variables are global among all front panel, HP-IB, and subroutine operations. A user-defined array or variable may be used in any HP 3245A command where a numeric parameter is required.

INTEGER Command Stored in Subroutines

If an **INTEGER** command is stored in a subroutine, the array name is defined immediately, but data storage for the array is not allocated until the subroutine is run. (The name cannot be used for a subroutine name, another array name, or a variable name.)

Mainframe Memory Limitations

Any number of arrays may be dimensioned up to the limit of available memory. Since all numeric arrays declared by an **INTEGER** command are integers, each element requires two bytes of memory for storage.

Recovering Memory Space

Use the **SCRATCH** command to recover memory space allocated for arrays by the **INTEGER** command. Following **SCRATCH**, any array name can be used.

Related Commands

DIM, DISP, FETCH, FILL, REAL, SIZE?, VREAD

Example

Example: Dimension INTEGER Variables and Arrays

These program line define an INTEGER array A with 10 elements (0 through 9) and an INTEGER variable B.

```
10  OUTPUT 709;"INTEGER A(9)"  !Define INTEGER array A with 10 elements
20  OUTPUT 709;"INTEGER B"     !Define INTEGER variable B
                .
                .
```

LET

Description

LET Assignment. Assigns values to numeric variables. The keyword **LET** is optional.

Syntax

[LET] ***variable = expression***

Parameters

variable

The *variable* parameter may be a numeric variable or an array name with a numeric expression index such as A(4) or A(I+4). All arrays must be dimensioned before use.

expression

Any valid number or numeric expression (enclosed in parentheses).

Remarks

Logical Expressions

Logical expressions (e.g. B=C, B>C, B<C, B≥C, B≤C) can be specified for the expression parameter. For example:

```
20    OUTPUT 709;"A=(B>C)"
```

will set A = 1 if B>C, or to 0 if B<C.

Reading Assigned Values

The values assigned to variables and expressions can be read using the **DISP**, **FETCH,** or **VREAD** commands.

Related Commands

DIM, DISP, FETCH, INTEGER, REAL, VREAD

Example

Example: Variable Declarations

This program evaluates the sine of 0.223 radians and displays the result on the controller CRT. Note that since the **LET** keyword is optional, line 10 can be replaced with `10 OUTPUT 709;"A = SIN(.223)"`.

```
10  OUTPUT 709;"LET A = SIN(.223)"  !Place sine of 0.223 into A
20  OUTPUT 709;"VREAD A"            !Read value of A into output buffer
30  ENTER 709; Reslt                !Enter result
40  PRINT Reslt                     !Display result
50  END
```

A typical return is:

```
    .22115633
```

Description

Logarithm. Returns the logarithm (base 10) of the argument.

Syntax

LGT (*argument*)

Parameters

argument

The *argument* parameter must be a number or numeric expression (enclosed in parentheses) greater than zero.

Remarks

Related Commands

EXP, LOG

Example

Example: Compute Logarithm, Base 10

This program computes the logarithm (base 10) of 15 and displays the result on the controller CRT.

```
10  OUTPUT 709;"VREAD LGT(15)"  !Compute and read log (base 10) of 15
20  ENTER 709;A                 !Enter result
30  PRINT "Logarithm = ";A      !Print result
40  END
```

A typical return is:

```
    Logarithm = 1.1760913
```

LIST

Description

List Subroutine. Lists the specified subroutine to the controller CRT or to the front panel display.

Syntax

LIST *sub_name*

Parameters

sub_name Name of the subroutine to be listed.

Remarks

Data Returned

The **LIST** command lists the specified subroutine one line at a time to the controller CRT or the front panel display. The last word returned is "SUBEND ; ".

Compressed Subroutines Cannot Be Listed

A subroutine which has been compressed (see **COMPRESS** command) cannot be listed using the **LIST** command.

Listing Subroutines from Front Panel

When listing a subroutine from the front panel, scroll through the menu until the LIST command appears in the display. Press the right arrow key to move the blinking cursor one position to the right. Press the up arrow or down arrow key until the name of the subroutine you want to list appears in the display. Press ENTER then the down arrow key to step through the subroutine.

Listing Format

Subroutines are listed one command per line, with a semicolon following each command. Commas that appear in a subroutine command are replaced with blank spaces. The last line of each listing is "SUBEND ;".

Data Destination

When **LIST** is executed from the front panel, the response is displayed on the front panel display. When **LIST** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

Related Commands

COMPRESS, SUB, SUBEND

Example

Example: Creating and Listing a Subroutine (LIST)

This program creates subroutine TEST1 and lists the subroutine on the controller CRT.

```
10    !file LIST
20    !
30    CLEAR 709                         !Clear 3245A
40    OUTPUT 709;"RST"                  !Reset 3245A
50    OUTPUT 709;"SCRATCH"              !Clear 3245A memory
60    !
70    OUTPUT 709;"SUB TEST1"            !Download subroutine
80    OUTPUT 709;"  USE 0"
90    OUTPUT 709;"    DELAY 1.5;APPLY DCV 0.1"
100   OUTPUT 709;"    APPLY DCV 0.2"
110   OUTPUT 709;"SUBEND"
120   !
130   OUTPUT 709;"LIST TEST1"           !List subroutine
140   REPEAT
150     ENTER 709;A$
160     PRINT A$
170   UNTIL A$="SUBEND ;"
180   END
```

A typical return is:

```
SUB TEST1 ;
USE 0 ;
DELAY 1.5 ;
APPLY DCV 0.1 ;
APPLY DCV 0.2 ;
SUBEND ;
```

LOCAL (LCL)

Description

Go to Local. **LOCAL** (or **LCL**) returns the HP 3245A to the local mode.

Syntax

LOCAL

Parameters

None.

Remarks

LOCAL Command Operation

The **LOCAL** command like the LOCAL key, returns the HP 3245A to the local (front panel) mode. Note, however, if the keyboard is locked (LOCK ON command), executing the LOCAL command will return the 3245A to the local mode, but the front panel will remain disabled. The LOCK OFF command (from the controller) is required to restore keyboard control.

Related Commands

LOCK

Example

Example: Enabling Local Mode

This program line enables the local (front panel) mode.

```
70  OUTPUT 709;"LOCAL"  !Enable local (front panel) mode.
```

LOCAL (GTL)

Description

Returns the HP 3245A to the local mode.

Syntax

LOCAL 7
LOCAL 709

Parameters

None.

Remarks

Using LOCAL 7 and LOCAL 709

Executing LOCAL 709 returns the 3245A (or the instrument at address 9) to the local (front panel) mode. When a command is sent over the HP-IB, the instrument returns to the remote mode. Executing LOCAL 7 returns all devices at select code 7 to their local modes. When commands are received over HP-IB, they execute, however; the instruments return to the local mode. To ensure that the instrument(s) remains in the remote mode when a command is received over HP-IB (because of LOCAL 7), execute the HP-IB command REMOTE 7 before you resume programming.

Examples

```
LOCAL 7       !Sets HP-IB REN line FALSE (all devices go to local).
              !Must execute REMOTE 7 to return to remote mode.

LOCAL 709     !Issues HP-IB GTL to device at address 09. Then,
              !executing any HP 3245A command over HP-IB or sending
              !REMOTE 709 returns the HP 3245A to remote mode.
```

LOCAL LOCKOUT (LLO)

Description

Disables the HP 3245A LOCAL key from restoring front panel control. LOCAL LOCKOUT performs the same function as the HP 3245A **RMT** command.

Syntax

LOCAL LOCKOUT 7

Remarks

Using LOCAL LOCKOUT

- If the 3245A is in the local mode when **LOCAL LOCKOUT 7** is sent, it remains in local mode until placed in remote by a command sent over HP-IB, or by REMOTE 709. Once in remote, the LOCAL key is disabled and there is no access to the 3245A functions from the front panel.

- Once the 3245A LOCAL key has been disabled by **LOCAL LOCKOUT 7**, executing **LOCAL 709** will return the instrument to local until a subsequent command sent over HP-IB (or REMOTE 709) returns the instrument to the remote mode. The LOCAL key remains disabled. **LOCAL 7** followed by **REMOTE 7** returns the 3245A to the local mode and enables the LOCAL key to restore front panel control from the remote mode.

Example

Using LOCAL LOCKOUT 7

The following program shows how LOCAL LOCKOUT 7 is used to disable the 3245A LOCAL key, and how LOCAL 709 and LOCAL 7 interact with a local lockout.

```
10   LOCAL LOCKOUT 7 !Sends local lockout to 3245A. The instrument
20                   !is still in local.
30   REMOTE 709      !3245A is placed in remote
40   LOCAL 709       !Returns 3245A to local. LOCAL key is still
50                   !disabled.
60   REMOTE 709      !3245A is returned to remote.
70   LOCAL 7         !3245A returns to local. LOCAL LOCKOUT 7 cancelled.
80   REMOTE 7        !3245A will return and remain in remote when
90                   !command over HP-IB (or REMOTE 709) is received.
100  REMOTE 709      !3245A is again placed in remote.
110                  !The LOCAL key can now restore front panel
120                  !control.
130  END
```

Description

Lock Front Panel Keyboard. Enables or disables the HP 3245A front panel keyboard.

Syntax

LOCK *mode*

Parameters

mode

The *mode* parameters follow. Power-on/reset *mode* = OFF.

mode	Description
OFF	Enables the front panel keyboard.
ON	Front panel is totally disabled. Commands cannot be entered from the front panel regardless of the state (local/remote) of the HP 3245A.

Remarks

Re-enabling Local Control

Once locked, the 3245A keyboard is re-enabled by sending:

```
OUTPUT 709;"LOCK OFF"
```

Sending "`OUTPUT 709;"LOCAL"` restores access to all 3245A functions from the front panel.

Related Commands

LOCAL

Example

Example: Disable Front Panel

This program line disables the front panel keyboard. In this mode, no commands can be entered from the front panel. Issue **LOCK OFF** followed by **LOCAL** to re-enable front panel operation.

```
10 OUTPUT 709;"LOCK ON"     !Disables front panel keyboard
```

LOG

Description

Natural Logarithm. Returns the natural logarithm (base e) of the specified argument.

Syntax

LOG (*argument*)

Parameters

argument

The *argument* parameter must be a number or numeric expression (enclosed in parentheses) greater than zero.

Remarks

Related Commands

EXP, LGT

Example

Example: Compute Logarithm, Base e

This program computes the natural logarithm (base e) of 2.345 and displays the result (.8522854) on the controller CRT.

```
10  OUTPUT 709;"VREAD LOG(2.345)"   !Compute natural log of 2.345
20  ENTER 709;A                     !Enter result
30  PRINT "Natural Log = ";A        !Display result
40  END
```

A typical return is:

```
    Natural Log = .8522854
```

Description

Memory Mode. Specifies a variable or array to store data from commands which return numeric results. The **MEM** command cannot store ASCII (string) results.

Syntax

```
      OFF
MEM   variable
      array_name [(start_index) ]
```

Parameters

OFF

The OFF parameter returns the memory output mode back to normal operation (results are not stored in memory). When a command is executed from the front panel, the response is displayed on the front panel display. When **MEM** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

variable

If the *variable* parameter is specified, the next numeric command result is stored in the specified variable. The memory mode is then turned off (**MEM OFF**). Variable names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore character ("_"), or the question mark ("?"). Variable names must not be the same as HP 3245A commands or parameters, or stored state names.

array_name

If the *array_name* parameter is specified, all subsequent numeric data is stored in the specified REAL or INTEGER array. Array names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore character ("_"), or the question mark ("?"). Array names must not be the same as HP 3245A commands or parameters, or stored state names.

start_index

The *start_index* parameter is used with the *array_name* parameter to specify a starting location of the array (i.e., store numeric results at a specific array element). The range is any integer from 0 to 32767. Default *start_index* = 0 (begin storing data at the first (0th) element of the array).

Remarks

Power-On Memory Mode

At power-on, the memory mode is disabled (**MEM OFF**). The memory mode is also disabled by the **CLR** command, the Clear key, or an HP-IB Device Clear message.

Memory Mode OFF

When the memory storage mode is OFF, numeric output data is placed into the output buffer where it can be sent to the controller. If the output buffer is disabled (**OUTBUF OFF**), any data in the output buffer is replaced by new data. If the output buffer is enabled (**OUTBUF ON**), subsequent data is appended to previous data in the buffer and is available for later retrieval.

Memory Mode ON

When the memory storage mode is ON, numeric output data is sent once to the specified variable or continuously to the specified array. If data is to be stored in an array, the array must be dimensioned using **DIM** or **REAL** for REAL arrays or **INTEGER** for INTEGER arrays.

Use VREAD to Output Memory Results

The **VREAD** command sends the variable or array value(s) to the output buffer. Always disable the memory mode (**MEM OFF**) before executing the **VREAD** command. Otherwise, **VREAD** attempts to write into memory and an error occurs.

Related Commands

CLR, DIM, INTEGER, OUTBUF, REAL, VREAD

Examples

Example: Storing Results in a Variable (MEM)

This program first creates a two-element array STATE1 and sets channel A to output 2.25 VAC with 50 Ω output impedance and the 2.5 V range. Then, **MEM STATE1** sets the HP 3245A such that all subsequent query commands return the data to array STATE1. Thus, **IMP?** returns the output impedance value (50) to element 0 in STATE1 and **RANGE?** returns the channel range (2.5) to element 1 in STATE1. Since an array is used, note that memory mode MUST be turned OFF (line 170) prior to executing the **VREAD** command.

```
10   !file MEM
20   !
30   REAL A(0:1)                  !Define controller array
40   CLEAR 709                    !Clear HP 3245A
50   OUTPUT 709;"RST"             !Reset HP 3245A
60   OUTPUT 709;"SCRATCH"         !Clear HP 3245A memory
70   OUTPUT 709;"REAL STATE1(1)"  !Define REAL array
80   !
90   OUTPUT 709;"USE 0"           !Use channel A
100  OUTPUT 709;"  APPLY ACV 2.25" !Output 2.25 Vac PP
110  OUTPUT 709;"  IMP 50"        !Set 50 ohm output impedance
120  OUTPUT 709;"  RANGE 2.5"     !Set 2.5 V range
130  !
140  OUTPUT 709;"MEM STATE1"      !Set memory mode ON.
150  OUTPUT 709;"  IMP?"          !Impedance result to STATE1(0)
160  OUTPUT 709;"  RANGE?"        !Range result to STATE1(1)
170  OUTPUT 709;"MEM OFF"         !Turn memory mode OFF
180  OUTPUT 709;"VREAD STATE1"    !Trans values in STATE1 to outbuf
190  ENTER 709;A(*)               !Enter values
200  PRINT A(*)                   !Display values
210  END
```

A typical return is:

```
    50      2.5
```

MEMAVAIL?

Description

Memory Available Query. Returns the size (in bytes) of the largest volatile memory block available in HP 3245A memory.

Syntax

MEMAVAIL?

Parameters

None.

Remarks

Data Returned

MEMAVAIL? returns two numbers: the first number is the largest block (number of bytes) of volatile memory available, and the second number is always 0.

Using HP 3245A Memory

HP 3245A volatile memory may be allocated for variable, array, and subroutine storage. As volatile memory blocks are created and purged, memory becomes fragmented into blocks. **MEMAVAIL?** returns the size of the largest block of volatile memory available. Use **SCRATCH** to restore all volatile memory.

Data Destination

If the **MEM** command is used, the values returned by the **MEMAVAIL?** command are stored in the specified variable or array. When **MEMAVAIL?** is executed from the front panel, the response is displayed on the front panel display. When **MEMAVAIL?** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

HP-IB Data Format

The HP 3245A returns numeric results in either ASCII or binary format (see the **OFORMAT** command). In the default ASCII format, **MEMAVAIL?** returns real results in 8-digit scientific notation. In the binary format, the **MEMAVAIL?** command returns real results in IEEE-754 64-bit notation.

Related Commands

CAT, SCRATCH

Example

Example: Determining Available Memory

This program returns the number of bytes of the largest block of volatile memory available in HP 3245A memory.

```
10  DIM A$[60]                 !Dimension controller array
20  OUTPUT 709;"MEMAVAIL?"     !Read volatile memory size
30  ENTER 709;A$               !Enter volatile memory size
40  PRINT A$                   !Display size (volatile, 0)
50  END
```

If all memory is available, a typical return is:

```
    8.0248000E+04, 0.
```

MOD

Description

Modulo. Returns the remainder portion of a division (contrast with **DIV**).

Syntax

numerator **MOD** *denominator*

Parameters

numerator — A number, numeric variable, math function, array element, or numeric expression (enclosed in parentheses).

denominator — Must be a value NOT equal to zero. May be a number, numeric variable, math function, array element, or numeric expression (enclosed in parentheses).

Remarks

Related Commands

DIV, INT

Example

Example: Using the MOD Function

This program divides 7 by 3 and displays the remainder portion of the division (1) on the controller CRT.

```
10  OUTPUT 709;"VREAD 7 MOD 3"   !Compute and read MOD value
20  ENTER 709;A                  !Enter result
30  PRINT "MOD = ";A             !Display result
40  END
```

A typical return is:

```
    MOD = 1
```

Description

Monitor Conditions. Displays (on the front panel display) command keywords entered over the HP-IB or the state of channel A or channel B.

Syntax

```
       HPIB
MON    STATE ch
       NONE
       OFF
```

Parameters

HPIB Displays command keywords (with parameters), as the HP 3245A receives and executes commands from the controller.

STATE *ch* Displays the state of the specified channel (channel A or channel B). The ch parameters are **0, CHANA, 100, and CHANB)**

NONE/OFF The NONE or OFF parameter disables the monitoring mode. With NONE or OFF set, commands entered are displayed, but the display will not scroll.

Remarks

MON Cancels Previous MON Commands

A **MON** command cancels any monitoring processes from previous **MON** commands (only one monitor mode is in effect at a time). For example, **MON HPIB; MON STATE 0** cancels monitoring of the HP-IB inputs.

Monitoring Channel State

MON STATE 0 or **MON CHANA** monitors channel A, while **MON STATE 100** or **MON CHANB** monitors channel B. To disable the monitor mode, execute **MON OFF** or **MON NONE**.

Entering Monitor Display Into the Controller

The **MON** command normally displays its results on the front panel. You can, however, use the **DISP?** command to enter the monitor display into the controller. See **DISP?** for details and examples.

Power-On Display

The power-on/reset/clear state is **MON STATE CHANA**. At power-on/reset, the front panel display is as shown. The table following the display describes the elements.

Power-On/Reset State Display

```
Arrow to channel A, 0.000000E+0DCV, FREQ 1000.000   1000.000,
DCOFF 0.000000E+0, DUTY 50.0, PANG 0.000, RANGE 1.0, ARANGE
ON, TERM FRONT, IMP 0, DCRES HIGH, TRIGMODE OFF, TRIGIN HIGH,
TRIGOUT OFF, SYNCOUT OFF, REFIN INT, REFOUT EXT, DELAY 0.04.
```

Information	Description
Arrow to channel A	Channel A is USE chan and is MON STATE chan
0.000000E+0DCV	0 V DCV output
FREQ 1000.000 1000.000	Output freq in Hz*
DCOFF 0.000000E+0	DC offset in Volts
DUTY 50.0	Duty cycle percentage
PANG 0.000	Phase angle**
RANGE 1.0	Voltage/current range
ARANGE ON	Autorange setting
TERM FRONT	Output terminal
IMP 0	Output impedance
DCRES HIGH	DC resolution mode
TRIGMODE OFF	Triggering mode
TRIGIN HIGH	Trigger input source
TRIGOUT OFF	Output trigger mode
SYNCOUT OFF	SYNC destination
REFIN INT	Ref freq input source
REFOUT EXT	Ref freq output dest
DELAY 0.04	Output delay in sec

* = second value applies to dual-freq mode only.
** = applies to synchronized mode only.

Related Commands

DISP?, MEAS, VERIFY

Examples

Example: Monitor HP-IB Commands

This program demonstrates the use of **MON HPIB** to display the HP 3245A commands sent over HP-IB from the controller.

```
10  OUTPUT 709;"MON HPIB"        !Monitor HP-IB commands
20  OUTPUT 709;"USE 0"           !Use channel A
30  OUTPUT 709;"APPLY DCV .1"    !Output 0.1 DCV
40  END
```

A typical display on the front panel display follows. Note that the rest of the message "V .1 ;" can be read by scrolling the display.

```
HP-IB:     APPLY DC
```

Example: Monitor Channel State

This program monitors the state of channel A (slot 0).

```
10  OUTPUT 709;"RST"              !Reset HP 3245A
20  OUTPUT 709;"USE 0"            !Use channel A
30  OUTPUT 709;"APPLY DCV .1"     !Output 0.1 DCV
40  OUTPUT 709;"MON STATE 0"      !Monitor channel A
50  END
```

A typical display on the front panel display follows. Note that the remainder of the message can be read by scrolling the display.

```
1.000000E-1DCV
```

NOT

Description

Inclusive-NOT. Returns a "0" or "1" based upon the logical complement of the specified argument.

Syntax

NOT *argument*

Parameters

argument

A number, numeric variable, math function, array element, or numeric expression (enclosed in parentheses).

Remarks

NOT Operation

The **NOT** command returns "0" or "1" as shown in the following truth table. Any non-zero value (positive or negative) is treated as a logic "0". Only zero is treated as a logic "1".

A	NOT A
0	1
1	0

Related Commands

AND, BINCMP

Example

Example: Using the NOT Function (NOT)

This program shows the data returned by the NOT function. When the subroutine is called, A is assigned the value of 1 and B is assigned the value of 0. With these values, A EXOR B returns a 1. Since this argument of the NOT function is not zero, the NOT function itself evaluates to zero and `THE COMPLEMENT OF 1 IS 0` is displayed on the display.

```
10    !file NOT
20    !
30    CLEAR 709                 !Clear HP 3245A
40    OUTPUT 709;"RST"          !Reset HP 3245A
50    OUTPUT 709;"SCRATCH"      !Clear HP 3245A memory
60    OUTPUT 709;"A=1"          !A = 1
70    OUTPUT 709;"B=0"          !B = 0
80    !
90    OUTPUT 709;"SUB COMP"                         !Define sub
100   OUTPUT 709;"  IF NOT (A EXOR B) THEN"         !NOT (A EXOR B)=0
110   OUTPUT 709;"    DISP'THE COMPLEMENT OF 0 IS 1'"
120   OUTPUT 709;"  ELSE"
130   OUTPUT 709;"    DISP'THE COMPLEMENT OF 1 IS 0'" !Disp msg
140   OUTPUT 709;"  END IF"
150   OUTPUT 709;"SUBEND"                           !End sub
160   !
170   OUTPUT 709;"CALL COMP"                        !Call sub
180   END
```

Description

Output Format. Selects the output format (ASCII or binary) for HP 3245A commands which return data.

Syntax

OFORMAT *format*

Parameters

format

The *format* parameter specifies ASCII or binary output format. The *format* parameters follow. Power-on *format* = ASCII.

format	Definition
ASCII	6-digit signed notation (integer results). 8-digit scientific notation (real results).
BINARY	16-bit, 2's complement notation (integer results). IEEE-754 64-bit notation (real results).

Remarks

ASCII Mode

In ASCII mode, multiple results from one command are separated by commas. A carriage return and line feed are sent after the final result. If **END** is ON, an EOI is sent with the line feed.

Binary Mode

The binary mode is useful only if your controller uses 16-bit integer and 64-bit IEEE-754 format for its internal binary representation. The **BLOCKOUT** command determines whether the IEEE-728 Block A header is sent before each output.

Related Commands

BLOCKOUT, END

Example

Example: Using Binary Output Format (OFORMAT)

This program shows one way to transfer a REAL and an INTEGER value to the controller via an I/O path when binary output format is set with **OFORMAT BINARY**. Note that **BLOCKOUT OFF** is set, so the IEEE-728 block A header is not sent. Also, note that **OFORMAT ASCII** resets the output mode to ASCII at the end of the program.

```
10   !file OFORMAT
20   !
30   REAL C                           !Define C as REAL variable
40   INTEGER D                        !Define D as INTEGER variable
50   ASSIGNPath to 709;FORMAT OFF   !Assign I/O path - format off
60   !
70   CLEAR 709                        !Clear HP 3245A
```

```
80   OUTPUT 709;"RST"                !Reset HP 3245A
90   OUTPUT 709;"OUTBUF ON"          !Set output buffer on
100  OUTPUT 709;"OFORMAT BINARY"     !Set binary output format
110  OUTPUT 709;"BLOCKOUT OFF"       !Delete IEEE-728 Block A header
120  !
130  OUTPUT 709;"REAL A"             !Define A as REAL variable
140  OUTPUT 709;"INTEGER B"          !Define B as INTEGER variable
150  OUTPUT 709;"A=-1.234567E+2"     !Assign value to A
160  OUTPUT 709;"B=2E+1"             !Assign value to B
170  !
180  OUTPUT 709;"VREAD A"            !Read value of A
190  OUTPUT 709;"VREAD B"            !Read value of B
200  ENTERPath;C                   !Enter value of A
210  ENTERPath;D                   !Enter value of B
220  PRINT "A =";C                   !Display value of A
230  PRINT "B =";D                   !Display value of B
240  !
250  OUTPUT 709;"OFORMAT ASCII"      !Set ASCII output format
260  END
```

A typical return is:

```
A =-123.4567
B = 20
```

Description

Inclusive-OR. Returns a "0" or "1" based upon the logical OR of the specified arguments.

Syntax

argument **OR** *argument*

Parameters

argument

A number, numeric variable, math function, array element, or numeric expression (enclosed in parentheses).

Remarks

OR Operation

OR returns "0" or "1" as shown. Any non-zero value (positive or negative) is treated as a logic "1". Only zero is treated as a logic "0".

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

Related Commands

AND, BINIOR

Example

Example: OR Statement in IF...END IF Loop

This program uses **OR** with an **IF...END IF** loop. First, "A OR B" is calculated. If either (or both) A are <>0, "TRUE" is displayed. If both A and B = 0, "FALSE" is displayed. The first time the subroutine is called, A = 3 and B = 2 so "TRUE" is returned. The second time the subroutine is called, A = 0 and B = 0, so "FALSE" is returned.

```
10   CLEAR 709                     !Clear HP 3245A
20   OUTPUT 709;"RST"              !Reset HP 3245A
30   OUTPUT 709;"SCRATCH"          !Clear HP 3245A memory
40   OUTPUT 709;"SUB ORLOOP"       !Start subroutine
50   OUTPUT 709;"  INTEGER A,B"    !Define A and B as INTEGERS
60   OUTPUT 709;"  IF A OR B THEN" !Start loop
70   OUTPUT 709;"    DISP 'TRUE'"  !Display "TRUE" if A OR B = 1
80   OUTPUT 709;"  ELSE"
90   OUTPUT 709;"    DISP 'FALSE'" !Display "FALSE" if A OR B = 0
100  OUTPUT 709;"  END IF"         !End loop
110  OUTPUT 709;"SUBEND"           !End subroutine
120  OUTPUT 709;"A=3;B=2"          !Define A and B
130  OUTPUT 709;"CALL ORLOOP"      !Call subroutine
140  WAIT 1                        !Wait 1 second
150  OUTPUT 709;"A=0;B=0"          !Redefine A and B
160  OUTPUT 709;"CALL ORLOOP"      !Call subroutine again
170  END
```

OUTBUF

Description

Output Buffer. Enables or disables the HP 3245A output buffer. Output buffer capacity is 2048 bytes.

Syntax

OUTBUF *mode*

Parameters

mode

The *mode* parameters follow. Power-on/reset *mode* = OFF.

mode	Description
OFF	Disables output buffer. When disabled, any new data output over HP-IB overwrites previous data in the buffer.
ON	Enables output buffer. When enabled, any new data output over HP-IB is appended to existing data in the buffer. When the output buffer is full (2048 bytes), command execution is suspended until data is read from the buffer by the controller.

Remarks

Output Buffer OFF

When the output buffer is disabled (**OUTBUF OFF**), any new command which generates data clears the buffer and writes the new data into the buffer. If the output buffer is disabled, the input buffer should also be disabled (**INBUF OFF**) to ensure that data corresponds to the command which generated it.

Output Buffer Storage Capacity

Output buffer capacity is 2048 bytes. When the output buffer is full, a subsequent command which generates data will not complete until there is room in the buffer. When the command does not complete, the HP-IB is held thus preventing the controller from entering data. As a result, a "deadlock" occurs between the HP 3245A and the controller. Deadlocks can be avoided by entering data as it becomes available.

Reading Data When Buffer is Empty

If the HP 3245A is addressed to talk when the output buffer is empty, all further HP-IB activity stops until the data becomes available. If no command or subroutine has been executed which will eventually generate output data, the HP-IB remains "busy" until the controller times out.

Clearing the Output Buffer

The output buffer is cleared by a **CLROUT**, **CLR**, or **RST** command or by an HP-IB Device Clear.

HP-IB Data Format

The HP 3245A returns numeric results in either ASCII or binary format (see the **OFORMAT** command). In the default ASCII format, the HP 3245A returns integer results in 6-digit signed notation and real results in 8-digit scientific notation. In the binary format, the HP 3245A returns integer results in 16-bit, 2's complement notation and real results in IEEE-754 64-bit notation.

Related Commands

CLR, CLROUT, INBUF, MEM

Example

Example: Enabling Output Buffer

This program enables the output buffer. The data returned by **ADDR?** and **MEMAVAIL?** are stored in the output buffer and read by the controller.

```
10    DIM B$[60]                        !Dimension controller array
20    OUTPUT 709;"RST"                  !Reset HP 3245A
30    OUTPUT 709;"OUTBUF ON"            !Enable output buffer
40    OUTPUT 709;"ADDR?"                !Query HP 3245A address
50    OUTPUT 709;"MEMAVAIL?"            !Query available memory
60    ENTER 709;A                       !Enter address
70    PRINT "Address =";A               !Display address
80    ENTER 709;B$                      !Enter available memory
90    PRINT "Memory Available =";B$     !Display available memory
100   END
```

A typical return is:

```
Address = 9
Memory Available = 6.9768000E+04, 0.0000000E+00
```

OUTPUT?

Description

Output Value Query. Returns the last programmed output value (voltage or current) from the **USE** channel.

Syntax

OUTPUT?

Parameters

None.

Remarks

Data Returned

The **OUTPUT?** command returns the last programmed output value (voltage or current) from the **USE** channel. **OUTPUT?** returns results in volts DC, amps DC, or AC volts peak-to-peak. For the **APPLY DCMEMV** or **APPLY DCMEMI** functions, **OUTPUT?** returns the value of the array element with the largest magnitude.

Data Destination

When **OUTPUT?** is executed from the front panel, the response is displayed on the front panel display. When **OUTPUT?** is executed from the controller, the response is sent to the output buffer in the default ASCII format. IF **MEM ON** is set, the value returned by **OUTPUT?** is stored in the specified variable or array.

HP-IB Data Format

The HP 3245A returns numeric results in either ASCII or binary format (see the **OFORMAT** command). In the default ASCII format, **OUTPUT?** returns real results in 8-digit scientific notation. In binary format, the **OUTPUT?** command returns real results in IEEE-754 64-bit notation.

Related Commands

APPLYs, USE

Example

Example: Reading Output Voltage Value (OUTPUTQ)

This program returns the programmed output value from channel A (.21346 V ac PP).

```
10   !file OUTPUTQ
20   !
30   CLEAR 709                          !Clear HP 3245A
40   OUTPUT 709;"RST"                   !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"               !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"                 !Use channel A
80   OUTPUT 709;"APPLY RPV -.21346"     !Output ramp waveform
90   OUTPUT 709;"OUTPUT?"               !Query channel A output value
100  ENTER 709;A                        !Enter result
110  PRINT A                            !Display result
120  END
```

Description

Set/Read Phase Angle. When the synchronized mode is set (**TRIGMODE ARMWF**), **PANG** sets the phase angle difference between channel outputs for sine, ramp, square, and arbitrary waveforms. **PANG?** returns the phase angle setting for the **USE** channel.

Syntax

PANG *degrees*

Parameters

degrees

Phase angle (phase difference between channel outputs) from -360 degrees to +360 degrees with 0.001 degree resolution. Zero degrees is defined as the positive-going, zero-crossing point of the sine, ramp, or square waveform (relative to the DC offset). For arbitrary waveforms, zero degrees is defined as the first of the 2048 points required to define the waveform. One complete waveform equals 360 degrees. Power-on/reset *degrees* = 0.

Remarks

Using PANG

The **PANG** command has no effect except when used in the synchronized mode (**TRIGMODE ARMWF**).

Query Command (PANG?)

PANG? returns the phase angle setting for the **USE** channel.

Related Commands

APPLYs, TRIGMODE, USE

Example

Example: Setting Phase Angle (PANG)

This program sets channels A and B to synchronously output 5 Vac PP sine waveforms with a 90° phase angle between the two waveforms. (Refer to the **PHSYNC** command for another method of generating synchronous waveforms.)

```
10   !file PANG
20   !
30   CLEAR 709                        !Clear HP 3245A
40   OUTPUT 709;"RST"                 !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"             !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"               !Use channel A
80   OUTPUT 709;"  TRIGMODE ARMWF"    !Set synch mode on ch A
90   OUTPUT 709;"  PANG 0"            !channel A phase angle = 0 deg
100  OUTPUT 709;"  REFOUT EXT"        !Freq Ref conn is destination
110  OUTPUT 709;"  TRIGIN TB1"        !TB1 is ch A trigger source
120  OUTPUT 709;"  APPLY ACV 5"       !Output 5 Vac PP sine wave
130  OUTPUT 709;"USE 100"             !Use channel B
140  OUTPUT 709;"  TRIGMODE ARMWF"    !Set synch mode on ch B
150  OUTPUT 709;"  PANG 90"           !channel B phase angle = 90 deg
160  OUTPUT 709;   REFIN EXT"         !Freq Ref conn is source
170  OUTPUT 709;"  TRIGIN TB1"        !TB1 is ch B trigger source
```

```
180  OUTPUT 709;"  APPLY ACV 5"      !Output 5 Vac PP sine wave
190  !
200  OUTPUT 709;"DRIVETB1 SGL"       !Trigger both chs simultaneously
210  END
```

Description

Pause Subroutine. Pauses a subroutine executed with the **RUN** command. Use the **STEP** or **CONT** command to resume subroutine execution. A subroutine executed with the **CALL** command cannot be paused.

Syntax

PAUSE

Parameters

None.

Remarks

Use PAUSE Only if Subroutine is RUN

PAUSE may be executed inside or outside of a subroutine. However, **PAUSE** cannot be used with called subroutines (see the **CALL** command) since the HP 3245A will not accept any commands from the controller while a called subroutine is executing. However, for a **RUN** subroutine, the HP 3245A releases the HP-IB and a **PAUSE**, **CONT**, or **STEP** command will be recognized and accepted.

Using the CONT and STEP Commands

Running a subroutine and allowing it to execute until it reaches a **PAUSE** statement is similar to stepping tha subroutine to that statement. In either case, **CONT** resumes execution, while **STEP** executes only the next command. Only one subroutine (the last one to be paused or stepped) can be continued.

Determining if Subroutine is Paused

The **PAUSED?** command returns a "1" or "0" to indicate if the current subroutine is paused. See **PAUSED?** for details.

Related Commands

CALL, CONT, PAUSED?, RUN, STEP, SUB, SUBEND

Example

Example: Pausing a Subroutine

This program stores subroutine PAUSESQR and executes it using **RUN**. When the subroutine executes, the square roots of 1 through 9 are displayed on the front panel.

```
10   OUTPUT 709;"INTEGER R"        !Define INTEGER variable
20   OUTPUT 709;"SUB PAUSESQR"     !Begin subroutine
30   OUTPUT 709;"FOR R=0 TO 9"     !Begin loop
40   OUTPUT 709;"  DISP SQR(R)"    !Display square root
50   OUTPUT 709;"  PAUSE"          !Pause execution
60   OUTPUT 709;"NEXT R"           !Next value
70   OUTPUT 709;"SUBEND"           !End subroutine
80   OUTPUT 709;"RUN PAUSESQR"     !Run subroutine
90   WAIT 1                        !Wait 1 second
100  FOR I=1 TO 9                  !Start continue loop
110    OUTPUT 709;"CONT"           !Continue execution
120    WAIT .5                     !Wait .5 seconds
130  NEXT I                        !Next value
140  END
```

PAUSED?

Description

Pause Query. Returns "1" if the current **RUN** subroutine is paused or "0" if the subroutine is running (or finished running). Only subroutines executed with the **RUN** command can be paused.

Syntax

PAUSED?

Parameters

None.

Remarks

Data Returned

The **PAUSED?** command returns "1" if the current **RUN** subroutine is paused or "0" if the subroutine is running (or finished running).

Data Destination

If **MEM** is used, the value returned by **PAUSED?** is stored in the specified variable or array. When **PAUSED?** is executed from the front panel, the response is displayed on the front panel display. When **PAUSED?** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

HP-IB Data Format

The HP 3245A returns numeric results in either ASCII or binary format (see the **OFORMAT** command). In the default ASCII format, **PAUSED?** returns integer results in 6-digit signed notation. In the binary format, **PAUSED?** returns integer results in 16-bit, 2's complement notation.

Related Commands

PAUSE, RUN, SUB, SUBEND

Example

Example: Reading Paused State

This program executes subroutine PAUSESQR with the **RUN** command and then sends the **PAUSED?** command (PAUSESQR is assumed to have been previously downloaded).

```
10  OUTPUT 709;"RUN PAUSESQR"          !Run subroutine PAUSESQR
20  OUTPUT 709;"PAUSED?"               !Query paused state
30  ENTER 709;A                        !Enter state
40  PRINT "Paused State (1=Paused) =";A  !Display state
50  END
```

If the subroutine is paused, a typical return is:

```
    Paused State (1=Paused) = 1
```

Description **Synchronized Output Mode**. Sets channels A and B for synchronized outputs.

Syntax **PHSYNC**

Parameters None.

Remarks

Parameters Set With PHSYNC

Using the **PHSYNC** command is equivalent to setting channels A and B to a set of parameters and executing a trigger to output the waveforms synchronously. Note that waveform parameters, such as amplitude, frequency, DC offset, etc. must still be set with the appropriate command. A list of the commands set by executing **PHSYNC** follows.

Command	Description
DRIVETB1 HIGH	Set TB1 HIGH (+5 V)
USE 0	Use channel A
TRIGMODE ARMWF	Set synchronized mode on ch A
REFOUT EXT	Ref freq dest is Freq Ref connector
TRIGIN TB1	Ch A input trigger source is TB1
USE 100	Use channel B
TRIGMODE ARMWF	Set synchronized mode on ch B
REFIN EXT	Ref freq input is Freq Ref connector
TRIGIN TB1	Ch B input trigger source is TB1
DRIVETB1 SGL	Trigger both channels simultaneously
Note: USE is returned to the channel specified when PHSYNC was executed.	

PHSYNC Operation

With **PHSYNC**, channels A and B are set for synchronous operation (with **TRIGMODE ARMWF**), the reference frequency is routed via the Freq Ref connector, and triggering is via the TB1 trigger bus. Thus, if these parameters can be used, **PHSYNC** offers a shorter method to enter the parameters required. (The waveform parameters must still be specified, however.)

After **PHSYNC** is executed, both channels remain in synchronized mode (**TRIGMODE ARMWF**). Subsequent execution of **APPLY**, **ARANGE ON**, **DCOFF**, **DUTY**, **FREQ**, **IMP**, **PANG**, **RANGE**, and **TRIGMODE ARMWF** commands will cause that channel's output to be suspended until a trigger is received on a channel (another **PHSYNC** will provide this trigger) or until a **TRIGMODE** other than **ARMWF** is selected. (Sending **ARANGE OFF**, **IMP**, or **RANGE** to the same range will not suspend a channel's output.)

Related Commands

REFIN, REFOUT, TRIGIN, TRIGMODE, USE

Example

Example: Generating Synchronized Waveforms (PHSYNC)

This program outputs a 5 kHz sine waveform (phase = 0°) from the channel A (the master channel) and a 5 kHz ramp waveform (phase = 180°) from channel B (the slave channel). **PHSYNC** sets synchronized waveform mode, sets TB1 as the channel input trigger sources, sets the Freq Ref connector as the output/input path for the reference frequency, and simultaneously triggers both channels (via TB1) to output their waveforms.

```
10   !file PHSYNC
20   !
30   CLEAR 709                        !Clear HP 3245A
40   OUTPUT 709;"RST"                 !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"             !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"               !Use channel A
80   OUTPUT 709;"  FREQ 5E3"          !Output freq is 5 kHz
90   OUTPUT 709;"  APPLY ACV 5"       !Output sine wave 5 Vac PP
100  OUTPUT 709;"USE 100"             !Use channel B
110  OUTPUT 709;"  FREQ 5E3"          !Output freq is 5 kHz
120  OUTPUT 709;"  APPLY ACV 5"       !Output ramp waveform 5 V
130  !
140  OUTPUT 709;"PHSYNC"              !Generate synced outputs
150  END
```

Description

Power-On SRQ. Enables or disables assertion of the HP-IB SRQ (service request) line at power-on. When enabled, the RQS mask (see the **RQS** command) is set to "8" at power-on thus allowing SRQ to be asserted. When disabled, the RQS mask is set to "0" the next time power is cycled.

Syntax

PONSRQ *mode*

Parameters

mode

The *mode* parameters follow. Power-on *mode* = most recent value stored in continuous memory. Factory setting = OFF.

mode	Description
ON	Enables power-on service request (SRQ).
OFF	Disables power-on service request (SRQ).

Remarks

Using PONSRQ

Use the **PONSRQ ON** command if the controller can detect and respond to the HP-IB SRQ line. The controller can then be programmed to take appropriate action if a power failure occurs.

PONSRQ Setting Stored in Continuous Memory

The **PONSRQ** setting (ON/OFF) is stored in continuous memory and is retained when power is removed from the HP 3245A.

Related Commands

RQS, STB?

Example

Example: Enabling Power-On Service Request (PONSRQ)

This program enables the HP-IB SRQ line to be asserted at power-on (sets the RQS mask to "8"). If the factory setting of **PONSRQ OFF** is assumed, "0" is returned when the program is first executed. However, if power is cycled and the program rerun, "8" is returned.

```
10   !file PONSRQ
20   !
30   OUTPUT 709;"PONSRQ ON"   !Enables HP-IB SRQ line assertion
40   OUTPUT 709;"RQS?"        !Query RQS mask state
50   ENTER 709;A              !Enter state
60   PRINT A                  !Display state
70   END
```

A typical return (after power is cycled) is:

8

RANGE/RANGE?

Description

Set/Read Output Range. **RANGE** selects a specific output range or enables the autorange function on the **USE** channel. **RANGE?** returns the current range setting on the **USE** channel.

Syntax

RANGE [*max_output*]

RANGE?

Parameters

max_output

To select a range, specify *max_output* as the maximum expected output voltage or current. The channel then selects the correct range, based on the previous mode (DCI, DCV, ACI, or ACV) of the **USE** channel. To enable the autorange function, use the word "AUTO" for *max_output* or default the parameter. When autoranging is enabled, the channel automatically selects the lowest voltage or current range which will provide maximum accuracy for the desired output level. Power-on/reset/default *max_output* = AUTO.

Remarks

DC Voltage/Current Ranges

The following table shows DC voltage and current ranges and maximum programmed output values. The values shown assume that the channel is terminated with a 50Ω load in the 50Ω mode. When a fixed range is selected, the output range is from 10% to 100% of the voltage range selected or from 5% to 100% of the current range selected.

DC Current Outputs

Range (Amps DC)	Maximum Programmed Output	Resolution High-Res Mode	Resolution Low-Res Mode
0.0001A	0.0001 A	0.1 nA	50 nA
0.001A	0.001 A	1 nA	500 nA
0.01A	0.01 A	10 nA	5 µA
0.1A	0.1 A	100 nA	50 µA

DC Voltage Outputs (High-Resolution Mode)

Output Impedance (Ohms)	Range (VDC)	Maximum Programmed Output	Resolution
0	1V	1.25 V	1 µV
0	10V	10.25 V	10 µV
50	0.5V	0.625 V	0.5 µV
50	5V	5.125 V	5 µV

DC Voltage Outputs (Low-Resolution Mode)

Output Impedance (Ohms)	Range (VDC)	Maximum Programmed Output	Resolution
0	0.15625V	0.15625 V	78 µV
0	0.3125V	0.3125 V	156 µV
0	0.625V	0.625 V	312 µV
0	1.25V	1.25 V	625 mV
0	2.5V	2.5 V	1.25 mV
0	5V	5 V	2.5 mV
0	10V	10 V	5 mV
50	0.078125V	0.078125 V	39 µV
50	0.15625V	0.15625 V	78 µV
50	0.3125V	0.3125 V	156 µV
50	0.625V	0.625 V	312 µV
50	1.25V	1.25 V	625 mV
50	2.5V	2.5 V	1.25 mV
50	5V	5 V	2.5 mV

Autorange Enabled When Output Parameters Change

Output range values differ according to DC resolution mode, output impedance, and output function. When any of these output parameters are changed, the channel enables autorange. When selecting the output ranges with **RANGE**, make the range selection AFTER the resolution mode, output impedance, and output function have been chosen. To avoid an "OUT OF RANGE" error, do not execute **RANGE** until you are sure the output can be generated on the range.

AC Voltage/Current Ranges

The following table shows AC voltage and current ranges and maximum programmed output values. The values shown assume that the channel is terminated with a 50Ω load in the 50Ω mode. When fixed range is selected, the output range is from 10% to 100% of the peak-to-peak range selected or from 5% to 100% of the current range selected.

AC Current Outputs

Range (Amps AC)	Maximum Programmed Output	Resolution
0.0002A	0.0002 A	50 nA
0.002A	0.002 A	500 nA
0.02A	0.02 A	5 µA
0.2A	0.2 A	50 µA

AC Voltage Outputs (Sine, Ramp, Square, Arb)			
Output Impedance (Ohms)	Range (Vac PP)	Maximum Programmed Output (Vac PP)	Resolution
0	0.3125V	0.3125 V	156 µV
0	0.625V	0.625 V	312 µV
0	1.25V	1.25 V	625 µV
0	2.5V	2.5 V	1.25 mV
0	5V	5 V	2.5 mV
0	10V	10 V	5 mV
0	20V	20 V	10 mV
50	0.15625V	0.15625 V	78 µV
50	0.3125V	0.3125 V	156 µV
50	0.625V	0.625 V	312 µV
50	1.25V	1.25 V	625 µV
50	2.5V	2.5 V	1.25 mV
50	5V	5 V	2.5 mV
50	10V	10 V	5 mV

Query Command (RANGE?)

The **RANGE?** command returns the value (in volts DC, amps DC, or volts or amps AC peak-to-peak) of the output on the **USE** channel (channel A or B).

Related Commands

APPLYs, ARANGE, DCRES, IMP, RANGE?, USE

Examples

Example: Selecting DC Voltage Range (RANGE_1)

This program sets channel A to the 10 VDC range. **RST** sets **DCRES HIGH, IMP 0, APPLY DCV**, and autorange on channel A. Then, **RANGE 8** selects the 10 VDC range and **APPLY DCV 1.25** outputs 1.25 VDC on the 10 VDC range. Since the output function is not changed from its power-on DCV function, **RANGE** is executed before **APPLY DCV**.

```
10   !file RANGE_1
20   !
30   CLEAR 709                   !Clear HP 3245A
40   OUTPUT 709;"RST"            !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"        !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"          !Use channel A
80   OUTPUT 709;"RANGE 8"        !Select 10 VDC range
90   OUTPUT 709;"APPLY DCV 1.25" !Output 1.25 VDC
100  END
```

Example: Selecting AC Voltage Range (RANGE_2)

This program sets channel A to the 20 VAC voltage range. Since **RST** sets **DCRES HIGH**, **IMP 0**, **APPLY DCV**, and autorange on channel A, the channel must be set to the ACV mode with **APPLY ACV** BEFORE **RANGE** is executed. If this is not done, an "OUT OF RANGE" error occurs, since the 20V range exceeds the maximum range (10V) for a channel set to the DC voltage mode.

```
10   !file RANGE_2
20   !
30   CLEAR 709                  !Clear HP 3245A
40   OUTPUT 709;"RST"           !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"       !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"         !Use channel A
80   OUTPUT 709;"APPLY ACV 4"   !Output 4 Vac 2.5 mV resolution
90   OUTPUT 709;"RANGE 20"      !Select 20 Vac range
100  END
```

READY?

Description

Ready Query. Returns "1" when the HP 3245A is ready to accept a new command. **READY?** is useful when the input buffer is enabled (**INBUF ON**).

Syntax

READY?

Parameters

None.

Remarks

Data Returned

READY? returns "1" when the HP 3245A is ready to accept new commands. If the HP 3245A is presently executing a command or subroutine, the instrument waits until the command or subroutine is complete before returning a "1" to indicate a ready state. The **READY?** command is useful only when the input buffer is enabled (**INBUF ON**). The HP 3245A remains "busy" until **READY?** completes.

Data Destination

If **MEM** is used, the value returned by **READY?** is stored in the specified variable or array. When **READY?** is executed from the front panel, the response is displayed on the front panel display. When **READY?** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

HP-IB Data Format

The HP 3245A returns numeric results in either ASCII or binary format (see the **OFORMAT** command). In the default ASCII format, **READY?** returns integer results in 6-digit signed notation. In the binary format, **READY?** returns integer results in 16-bit, 2's complement notation.

Related Commands

INBUF, SERIAL, SETTLE

Example

Example: Reading the Ready Status (INBUF)

This program shows how to use the **READY?** command to indicate when the input buffer is empty. When the program executes, the HP 3245A is reset and cleared and the input buffer is enabled (line 70). Then, a command string (line 80) with **READY?** as the last command is sent to the HP 3245A. As the commands are executing, `Buffer Commands Executing` is displayed. When **READY?** executes, `1` is returned to the output buffer and is entered into the controller. At that point, `Buffer Commands Complete` is displayed.

```
10   !file INBUF
20   !
30   CLEAR 709                          !Clear HP 3245A
40   OUTPUT 709;"RST"                   !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"               !Clear HP 3245A memory
60   !
70   OUTPUT 709;"INBUF ON"              !Enable input buffer
80   OUTPUT 709;"USE 0;DELAY 1.5;APPLY DCV 1.0;APPLY DCV 2.0;READY?"
90   DISP "Buffer Commands Executing"   !Display message
100  ENTER 709;A                        !Returns "1"
110  IF A=1 THEN
120    DISP "Buffer Commands Complete"  !Display message
130  END IF
140  END
```

REAL

Description

Dimension REAL Array or Variable. Reserves memory space in the HP 3245A to store real variables or arrays.

Syntax

REAL *name* [(*max_index*)] [,*name* [(*max_index*)], ...]

Parameters

name

Variable or array name. Variable or array names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore character ("_"), or the question mark ("?"). Variable or array names must not be the same as HP 3245A commands or parameters, or stored state names.

The *name* parameter specifies the name of an REAL array or variable. Use *name* (*max_index*) to dimension an REAL array [e.g., `REAL TEST1 (9)`]. To dimension a REAL variable, omit the *max_index* parameter (e.g., `REAL TEST1`).

max_index

The *max_index* parameter specifies the number of elements in the array. The valid range is 0 to 32767. Since the lower bound (option base) for all arrays is zero, the number of elements in the array is one more than *max_index*. For example, "REAL A(5)" reserves memory space for REAL array A with six elements (0 through 5).

Remarks

Redimensioning Arrays

Executing **REAL** defines a REAL array and fills all elements with zeroes. (Arrays may be redimensioned with a **REAL** or a **DIM** command.) The redimensioned array is initialized to zeroes in all elements (if the array is redimensioned to the same size, it is cleared and all previous data is lost).

User-Defined Arrays are Global

All REAL arrays and variables are global among all front panel, HP-IB, and subroutine operations. A user-defined array or variable may be used in any HP 3245A command where a numeric parameter is required.

REAL Command Stored in Subroutines

If a **REAL** command is stored in a subroutine, the array name is defined immediately, but data storage for the array is not allocated until the subroutine is run. (The name cannot be used for a subroutine name, another array name, or a variable name.)

REAL Command vs. DIM Command

The **REAL** command defines REAL arrays and REAL variables. The **DIM** command defines REAL arrays only.

Memory Limitations

Any number of arrays may be dimensioned up to the limit of available memory. Since all numeric arrays declared by a **REAL** command are REAL, each element requires eight bytes of memory for storage. (For REAL arrays, ERROR 41 - OUT OF MEMORY is generated if you try to define an array greater than about 10000 elements.)

Recovering Memory Space

Use the **SCRATCH** command to recover memory space allocated for arrays by the **REAL** command. Following **SCRATCH**, any array or variable name can be used.

Related Commands

DIM, DISP, FETCH, FILL, INTEGER, SIZE?, VREAD

Example

Example: Dimension REAL Variables and Arrays

These program lines define a REAL array A with 10 elements (0 through 9) and a REAL variable B.

```
10  OUTPUT 709;"REAL A(9)"  !Define REAL array A with 10 elements
20  OUTPUT 709;"REAL B"     !Define REAL variable B
              .
              .
```

REFIN/REFIN?

Description

Set/Read Reference Frequency Input. REFIN selects the reference frequency input source for the slave channel. **REFIN?** returns the reference frequency input source for the slave channel.

Syntax

REFIN *source*

REFIN?

Parameters

source Specifies the source from which the slave channel will receive its reference frequency. The *source* parameters follow. Power-on/reset *source* = INT.

source	Definition
INT	Internal. Use the channel's internal 1.07 MHz signal as the reference frequency input source.
EXT	Freq Ref Connector. Use the signal on the Freq Ref connector as reference frequency input source.
TB0	Trigger Bus 0. Use the signal on trigger bus 0 (TB0) as the reference frequency input source.
TB1	Trigger Bus 1. Use the signal on trigger bus 1 (TB1) as the reference frequency input.

Remarks

Synchronizing Multiple AC Waveforms

For synchronized outputs, one channel is designated as the "master" and the other channel is designated as the "slave" channel. Each channel (master or slave) generates a reference frequency (1073741.824 Hz) based on its internal reference oscillator. When **REFIN/REFOUT** are used, the slave channel's oscillator locks onto the reference frequency generated by the master channel.

When the master and slave channels are triggered (both channels must be triggered at the same time), each channel outputs its specified waveform (sine, ramp, square, or arbitrary) at its selected frequency and phase. The **REFIN** command selects the source from which the slave channel's reference oscillator will accept the reference frequency.

Changing Master/Slave Relationship (REFIN/REFOUT)

At power-on/reset, channel A is set to **REFOUT EXT; REFIN INT** and channel B is set to **REFOUT OFF;REFIN EXT**. This setting "slaves" channel B reference to channel A reference, making channel A the "master". However, when **REFOUT** is changed to **EXT** on a channel, it forces the other channel (if **REFOUT EXT** is set) to **REFOUT OFF;REFIN EXT**. That is, the master/slave relationship is reversed.

NOTE

When two HP 3245As have their Freq Ref terminals connected, at power-on they both drive the connector since **REFOUT EXT** *is selected at power-on. For proper operation, both channels on one of the HP 3245As should be set to* **REFIN EXT**.

REFIN Restrictions

Since a channel cannot be a master and a slave at the same time, whenever **REFIN** is executed with any parameter other than **INT**, the channel output source is automatically disabled (**REFOUT OFF**). Executing **REFOUT EXT** on a channel changes the other channel setting from **REFOUT EXT** to **REFIN EXT** (and, hence, **REFOUT OFF**).

Selecting REFIN EXT

When the slave channel is set to **REFIN EXT** and the master channel is set to **REFOUT EXT**, no type of external connection is required for the reference frequency, since the reference is sensed internally by the slave channel.

Query Command (REFIN?)

The **REFIN?** command returns the specified source (INT, EXT, TB0, or TB1) for the reference input frequency on the slave channel.

Related Commands

FREQ, PANG, PHSYNC, REFOUT, TRIGIN, TRIGMODE, USE

Example

Example: Selecting Ref Frequency Input Source (REFIN_OUT)

This program uses **REFIN** and **REFOUT** to lock channel B (slave) output to channel A (master) output. Both channels output a 5 V ac PP ramp waveform. Since the Trigger ports of the two channels are externally connected, a trigger on channel A is also routed to channel B so that both channels are simultaneously triggered.

Since **REFOUT EXT** and **REFIN EXT** are set, the reference frequency on channel A is routed (internally) via the Freq Ref connector on the rear panel (no external connection is required). This prevents any drift in relative frequency and ensures that the two channel outputs are exactly at the same frequency. **TRIGIN LOW** triggers both channels simultaneously to begin synchronous outputs.

```
10   !file REFIN_OUT
20   !
30   !External connections - connect a BNC cable between the
40   !Trigger (I/O) ports of channels A and B.
50   !
60   CLEAR 709                     !Clear HP 3245A
70   OUTPUT 709;"RST"              !Reset HP 3245A
80   OUTPUT 709;"SCRATCH"          !Clear HP 3245A memory
```

```
90   !
100  OUTPUT 709;"USE 0"             !Use channel A
110  OUTPUT 709;"  TRIGMODE ARMWF"   !Set synch mode on ch A
120  OUTPUT 709;"  TRIGOUT EXT"      !Enable trigger out from ch A
130  OUTPUT 709;"  REFOUT EXT"       !Freq Ref conn is destination
140  OUTPUT 709;"  FREQ 5E3"         !Output freq is 5 kHz
150  OUTPUT 709;"  APPLY RPV 5"      !Output 5 V ac PP ramp wave
160  !
170  OUTPUT 709;"USE 100"            !Use channel B
180  OUTPUT 709;"  TRIGMODE ARMWF"   !Set synch mode on ch B
190  OUTPUT 709;"  TRIGIN EXT"       !Trigger conn is trigger source
200  OUTPUT 709;   REFIN EXT"        !Freq Ref conn is input source
210  OUTPUT 709;"  FREQ 5E3"         !Output freq is 5 kHz
220  OUTPUT 709;"  APPLY RPV 5"      !Output 5 V ac PP ramp wave
230  !
240  OUTPUT 709;"TRIGIN LOW"         !Trigger both chs simultaneously
250  END
```

Description

Set/Read Reference Frequency Output. When the synchronized waveform mode is set (with **TRIGMODE ARMWF**), **REFOUT** selects the reference frequency output destination for the master channel. **REFOUT?** returns the reference frequency output destination from the master channel.

Syntax

REFOUT *destination*

REFOUT?

Parameters

destination

Specifies the destination from which the master channel will output its reference frequency. The *destination* parameters follow. Power-on/reset *destination* (channel A) = EXT.

destination	Definition
OFF	Disable reference frequency output from master channel. Places Freq Ref connector in high-impedance state.
EXT	Freq Ref Connector. Output reference frequency to the Freq Ref connector.
TB0	Trigger Bus 0. Output reference frequency from master channel to trigger bus 0.
TB1	Trigger Bus 1. Output reference frequency from master channel to trigger bus 1.

Remarks

Synchronizing Multiple AC Waveforms

In synchronized mode, one channel is designated as the "master" channel, while the other channel is designated as the "slave" channel. Each channel (master or slave) generates a reference frequency (nominally 1073741.824 Hz) based on its internal reference oscillator. In synchronized mode, the slave channel oscillator locks onto the reference frequency generated by the master channel oscillator.

When the master and slave channels are triggered (both channels must be triggered at the same time), each channel outputs its specified waveform (sine, ramp, square, or arbitrary) at its selected frequency and phase. The **REFOUT** command selects the destination from which the master channel will output its reference frequency.

Changing Master/Slave Relationship (REFIN/REFOUT)

At power-on/reset, channel A is set to **REFOUT EXT; REFIN INT** and channel B is set to **REFOUT OFF;REFIN EXT**. This setting "slaves" channel B reference to channel A reference, making channel A the "master". However, when **REFOUT** is changed to **EXT** on a channel, it forces the other channel (if **REFOUT EXT** is set) to **REFOUT OFF;REFIN EXT**. That is, the master/slave relationship is reversed.

NOTE

When two HP 3245As have their Freq Ref terminals connected, at power-on they both drive the connector since **REFOUT EXT** *is selected at power-on. For proper operation, both channels on one of the HP 3245As should be set to* **REFIN EXT**.

REFOUT/SYNCOUT/DRIVETBn Interaction

Using **REFOUT TBn** or **SYNCOUT TBn** automatically resets **DRIVETBn** to **OFF**. Using **DRIVETBn** other than **OFF**, **SYNCOUT TBn**, or **REFOUT TBn** from the other channel automatically resets **REFOUT TBn** on this channel to **REFOUT OFF**.

REFOUT Restrictions

Since a channel cannot be a master and a slave at the same time, whenever **REFOUT** is executed with any parameter other than **OFF**, the channel input source is automatically set to internal (**REFIN INT**). Executing **REFOUT EXT** on a channel will change the other channel setting from **REFOUT EXT** to **REFIN EXT**.

Query Command (REFOUT?)

The **REFOUT?** command returns the reference frequency output destination (OFF, EXT, TB0, or TB1) for the master channel.

Related Commands

DRIVETBn, FREQ, PANG, PHSYNC, REFIN, SYNCOUT, TRIGIN, TRIGMODE, USE

Example

Example: Selecting Ref Freq Output Destination (REFIN_OUT)

This program uses **REFIN** and **REFOUT** to lock channel B (slave) output to channel A (master) output. Both channels output a 5 V ac PP ramp waveform. Since the Trigger ports of the two channels are externally connected, a trigger on channel A is also routed to channel B so that both channels are simultaneously triggered.

Since **REFOUT EXT** and **REFIN EXT** are set, the reference frequency on channel A is routed (internally) via the Freq Ref connector on the rear panel (no external connection is required). This prevents any drift in relative frequency and ensures that the two channel outputs are exactly at the same frequency. Note that a single trigger (**TRIGIN LOW**) triggers both channels simultaneously to begin synchronous outputs.

```
10   !file REFIN_OUT
20   !
30   !External connections - connect a BNC cable between the
40   !Trigger (I/O) ports of channels A and B.
50   !
60   CLEAR 709                        !Clear HP 3245A
70   OUTPUT 709;"RST"                 !Reset HP 3245A
80   OUTPUT 709;"SCRATCH"             !Clear HP 3245A memory
90   !
100  OUTPUT 709;"USE 0"               !Use channel A
110  OUTPUT 709;"  TRIGMODE ARMWF"    !Set synch mode on ch A
120  OUTPUT 709;"  TRIGOUT EXT"       !Enable trigger out from ch A
130  OUTPUT 709;"  REFOUT EXT"        !Freq Ref conn is destination
140  OUTPUT 709;"  FREQ 5E3"          !Output freq is 5 kHz
150  OUTPUT 709;"  APPLY RPV 5"       !Output 5 V ac PP ramp wave
160  !
170  OUTPUT 709;"USE 100"             !Use channel B
180  OUTPUT 709;"  TRIGMODE ARMWF"    !Set synch mode on ch B
190  OUTPUT 709;"  TRIGIN EXT"        !Trigger conn is trigger source
200  OUTPUT 709;   REFIN EXT"         !Freq Ref conn is input source
210  OUTPUT 709;"  FREQ 5E3"          !Output freq is 5 kHz
220  OUTPUT 709;"  APPLY RPV 5"       !Output 5 V ac PP ramp wave
230  !
240  OUTPUT 709;"TRIGIN LOW"          !Trigger both chs simultaneously
250  END
```

REMOTE

Description

Sets the HP-IB REN line TRUE.

Syntax

REMOTE 7
REMOTE 709

Remarks

- The **REMOTE 709** command places the HP 3245A in the remote state. However, **REMOTE 7** does not, by itself, place the HP 3245A in the remote state. After receiving **REMOTE 7**, the HP 3245A will only go to the remote state after receiving its listen address.

- Generally, the **REMOTE** command is required only after the **LOCAL** command is used. **REMOTE** is independent of any other HP-IB activity and is sent on the REN line. Most controllers set the REN line TRUE when power is applied or when the HP 3245A is reset.

Example

```
REMOTE 7       !Sets HP-IB REN line TRUE but does not place
               !any device in remote state.

REMOTE 709     !Sets HP-IB REN line TRUE and addresses device
               !at address 22 which places device in remote state.
```

RESET (RST)

Description

Reset Instrument/Channel. Resets the HP 3245A or the specified **USE** channel. **RESET** (or **RST**) is the same as **CRESET**.

Syntax

RESET [*ch*]

Parameters

ch Channel identifier. Use **0** or **CHANA** for channel A, use **100** or **CHANB** for channel B. Executing **RESET** without the *ch* parameter resets the entire HP 3245A. Executing **RESET** *ch* resets only the specified channel.

Remarks

RESET Command Action

- Performs all actions of device clear (See **CLR**).
- Disables input/output buffering and EOI function.
- Enables block output mode (**BLOCKOUT ON**).
- Trigger buses returned to power-on state.
- Both channels returned to reset state.
- Immediately aborts all running commands.

HP 3245A Power-On/Reset State

RESET returns the HP 3245A to the reset state shown in the following table. The reset state is the same as the power-on state except that the HP-IB addressed state is not changed (if in LOCAL,it remains in LOCAL or, if in REMOTE it remains in REMOTE); the power-on self-test is not performed; and user variables, arrays, and subroutines are not purged.

Item	Command	Power-On/Reset State
Output Function	APPLY	DC Volts (0 V)
Auto Range	ARANGE	On
DC Offset	DCOFF	0 Volts
DC Resolution	DCRES	High-Res Mode
Delay Time	DELAY	0.04 Seconds
Drive TB0	DRIVETB0	Off
Drive TB1	DRIVETB1	Off
Duty Cycle	DUTY	50%
Frequency	FREQ	1000 Hz
Output Impedance	IMP	0 Ohms
Phase Angle	PANG	0 Degrees
Range	RANGE	1V (Autogange)
Ref Freq Input	REFIN	External (ch B)
Ref Freq Output	REFOUT	External (ch A)
Sync Destination	SYNCOUT	Off
Output Terminal	TERM	Front
Input Trigger Source	TRIGIN	High
Trigger Mode	TRIGMODE	Off
Trigger Output Mode	TRIGOUT	Off

Related Commands

CLR, CRESET

Examples

Example: Using RESET

Line 10 returns the HP 3245A (including channels A and B) to its reset state, while line 100 resets channel A only.

```
10   OUTPUT 709;"RESET"      !Reset HP 3245A
                .
                .
100  OUTPUT 709;"RST 0"      !Reset channel A
                .
                .
```

Description

Return to Caller. Returns program execution from a called subroutine to the program line following the **CALL** statement. For subroutines executed using **RUN**, **RETURN** terminates execution of the subroutine.

Syntax

RETURN

Parameters

None.

Remarks

RETURN Valid Only Within Subroutines

RETURN can only be used within subroutines. It returns control to the caller at any point in a subroutine without executing the **SUBEND** command. Since **SUBEND** returns control to the calling program at the end of a subroutine, **RETURN** is only useful inside an **IF..END IF** loop.

Related Commands

IF..END IF, SUB, SUBEND

Example

Example: Using the RETURN Statement (RETURN)

This program shows how **RETURN** is used to exit a subroutine when a specific condition occurs. When the program executes, subroutine RANG_CHK is downloaded and applies DC voltages from 0.5 to 1.5 VDC in 0.1 VDC increments.

As each DC level is applied, the output range is read (line 130) and stored in variable A (line 100). As long as the output remains on the 1V range, the program continues. When the output range changes to the 10V range (1.25 VDC), the subroutine returns to the first line following the **CALL** statement (line 200).

Then, memory mode is turned off and **VREAD** reads the range value into the output buffer and the value is entered into the controller and `RANGE CHANGE HAS OCCURRED` is displayed. Meanwhile, the HP 3245A will have output voltages up to 1.3 VDC and the output will remain at 1.3 VDC. (Note that 1.3 VDC is applied before the range is checked.)

```
10   !file RETURN
20   !
30   CLEAR 709                             !Clear HP 3245A
40   OUTPUT 709;"RST"                      !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"                  !Clear HP 3245A memory
60   !
70   OUTPUT 709;"SUB RANG_CHK"             !Begin subroutine
80   OUTPUT 709;"  USE 0"                  !Use channel A
90   OUTPUT 709;"  FOR I = .5 TO 1.5 STEP 1"  !Set DCV amplitudes
100  OUTPUT 709;"    MEM A"                !Enable memory mode
110  OUTPUT 709;"    APPLY DCV I"          !Output DC levels
120  OUTPUT 709;"    WAIT 2"               !Wait 2 seconds
130  OUTPUT 709;"    RANGE?"               !Query range - store in A
140  OUTPUT 709;"    IF A<>1 THEN"         !Begin comparison loop
```

```
150  OUTPUT 709;"      RETURN"                !Exit sub if range <>1V
160  OUTPUT 709;"    END IF"                  !End loop
170  OUTPUT 709;"  NEXT I"                    !Increment count
180  OUTPUT 709;"SUBEND"                      !End subroutine
190  OUTPUT 709;"CALL RANG_CHK"               !Call subroutine
200  OUTPUT 709;"MEM OFF"                     !Turn memory mode off
210  OUTPUT 709;"VREAD A"                     !Read range value
220  ENTER 709;A                              !Enter range value
230  IF A <>1 THEN                            !Begin comparison loop
240    PRINT "RANGE CHANGE HAS OCCURRED"      !Display if range <>1
250  END IF                                   !End loop
260  END
```

Description

Revision Query. Returns the HP 3245A firmware revision date code.

Syntax

REV?

Parameters

None.

Remarks

Data Returned

REV? returns a four-digit year and date code. The code has the form "yyww", where "yy" is the year minus 1960, and "ww" is the week number of that year. For example, 2833 means that the latest firmware revision was the 33rd week of 1988 (1988 - 1960 = 28).

Data Destination

If **MEM** is used, the value returned by **REV?** is stored in the specified variable or array. When **REV?** is executed from the front panel, the response is displayed on the front panel display. When **REV?** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

HP-IB Data Format

The HP 3245A returns numeric results in either ASCII or binary format (see the **OFORMAT** command). In the default ASCII format, **REV?** returns integer results in 6-digit signed notation. In the binary format, **REV?** returns integer results in 16-bit, 2's complement notation.

Example

Example: Reading Firmware Revision Code (REVQ)

This program reads the firmware revision date of the HP 3245A and displays the result on the controller.

```
10   !file REVQ
20   !
30   CLEAR 709                  !Clear HP 3245A
40   OUTPUT 709;"RST"           !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"       !Clear HP 3245A memory
60   !
70   OUTPUT 709;"REV?"          !Read firmware date
80   ENTER 709;A                !Enter result
90   PRINT A                    !Display result
100  END
```

A typical return is:

```
2833
```

RMT

Description

Remote. Executes a soft local lockout (similar to HP-IB LOCAL LOCKOUT) which disables the front panel **Local** key while in remote mode.

Syntax

RMT

Parameters

None.

Remarks

RMT Command Operation

Normally, when the HP 3245A is in remote (i.e., commands are being sent from the controller), the **Local** key may be used to return to the local mode (i.e., front panel has full access). However, when **RMT** is executed, it prevents the front panel **Local** key from returning the HP 3245A to the local mode.

Gaining Front Panel Access after RMT

To enable the front panel **Local** key, execute the **LOCAL** command from the controller (e.g., `OUTPUT 709;"LOCAL"`).

Disabling Front Panel Keyboard and Display

Use the **LOCK ON** command to disable all keys on the front panel keyboard. Use the **DISP OFF** command to disable the front panel display.

Related Commands

DISP, LCL, LOCAL, LOCK

Example

Example: Disabling the Local Key

This program line disables the front panel **Local** key. To reenable the key, send the **LOCAL** command from the controller.

```
10  OUTPUT 709;"RMT"   !Disable front panel Local key
```

Description

Rotate Bits with Wraparound. Returns an integer obtained by rotating the argument a specified number of positions with bit wraparound (contrast with **SHIFT**).

Syntax

ROTATE (*argument*, *bit_displacement*)

Parameters

argument

Must be a numeric integer within the range -32768 to +32767. Within the HP 3245A, the argument is represented as a 16-bit, 2's complement integer.

bit_displacement

Specifies the number of positions the bits are rotated. Positive values rotate the argument toward the least significant bit and negative values rotate the argument toward the most significant bit.

Remarks

Related Commands

SHIFT

Examples

Example: Positive Rotation

This program rotates the bit pattern for 12 three positions toward the least significant bit and displays the result (-32767) on the controller CRT.

```
         12 = 0000 0000 0000 1100
Rotation #1 = 0000 0000 0000 0110
Rotation #2 = 0000 0000 0000 0011
Rotation #3 = 1000 0000 0000 0001
```

```
10  OUTPUT 709;"VREAD ROTATE(12,3)"  !Rotate bit pattern 3 pos
20  ENTER 709;A                      !Enter result
30  PRINT "Rotation Result = ";A     !Display result
40  END
```

Example: Negative Rotation

This program rotates the bit pattern for 12 three positions toward the most significant bit and displays the result (96) on the controller CRT.

```
         12 = 0000 0000 0000 1100
Rotation #1 = 0000 0000 0001 1000
Rotation #2 = 0000 0000 0011 0000
Rotation #3 = 0000 0000 0110 0000
```

```
10  OUTPUT 709;"VREAD ROTATE(12,-3)  !Rotate bit pattern 3 pos
20  ENTER 709;A                      !Enter result
30  PRINT "Rotation Result = "; A    !Display result
40  END
```

Description

Set/Read Service Request Enable. RQS sets bit(s) in the RQS (service request) mask register to determine which condition(s) will assert the HP-IB SRQ line. **RQS?** returns a decimal value equal to the sum of the values of the unmasked bits in the RQS mask.

Syntax

RQS *unmask_value*

Parameters

unmask_value

A decimal value equal to the sum of the values of the bits to be "unmasked". Only bits 0, 2, 3, 4, and 5 can be unmasked (bit 6 is always unmasked and bit 2 is not used). Other bits are ignored. Bit numbers, decimal value, and descriptions for the RQS mask are the same as for the Status Register.

Remarks

RQS mask Operation

The RQS mask determines which events and/or conditions monitored by the Status Register set Status Register bit 6 true (1) and thus assert the HP-IB SRQ line to the controller. If an event or condition monitored by the Status Register occurs and the corresponding bit in the RQS mask is "unmasked", the HP 3245A sets Status Register bit 6 (SRQ) true, asserts HP-IB SRQ and turns the front panel SRQ annunciator on. (The controller must be programmed to respond to the SRQ interrupt.)

Status Register Bit Definitions

The HP 3245A Status Register constantly monitors several predefined events and conditions. Whenever a defined event or condition occurs, the corresponding Status Register Bit is set true (1). If the bit is unmasked by **RQS**, an HP-IB SRQ is sent to the controller. Status Register bit definitions and associated decimal values used to unmask the bits follow.

Bit	Value	Definition	Set by:	Cleared by:
0	1	DATA AVAILABLE	Data available in the output buffer.	CLR, CLEAR, CLROUT, or Clear key or when data is removed from output buffer
1	2	NOT USED		
2	4	USER SERVICE REQUEST	SRQ command from controller or from HP 3245A.	STA?, CLR, CLEAR, or Clear key.
3	8	LOCAL	Power-on, RST, or Local mode set.	STA?, CLR, CLEAR, or

				Clear key.
4	16	READY	Input buffer is empty and no command or CALL subroutine is executing.	Command or CALL subroutine is executing.
5	32	ERROR	An error condition occurring.	CLR, CLEAR or when ERR? or ERRSTR? clears error register.
6	64	SRQ SENT	When any unmasked bit (0, 2, 3, 4, or 5) in status register is set.	STA?, STB?, CLR, CLEAR, SPOLL, or Clear key.*
7-16		NOT USED		

* = Not cleared by STA?/STB? for every condition.

Power-On Condition

If **PONSRQ OFF** is specified, the power-on value is 0. If **PONSRQ ON** is specified, the power-on value is 8, causing SRQ to be asserted at power-on.

Query Command (RQS?)

The **RQS?** command returns the decimal value of the sum of unmasked bits in the RQS mask register.

Related Commands

RQS?, PONSRQ, SRQ, STA?, STB?

Examples

Example: Unmasking RQS Mask Bit

This program unmasks RQS mask bit 3 (LOCAL). The program then continuously loops (at line 50) until the front panel **Local** key is pressed. When the **Local** key is pressed, the HP 3245A reverts to local mode and "HP 3245A NOW IN LOCAL" is displayed on the controller CRT.

```
10   CLEAR 709                          !Clear HP 3245A
20   OUTPUT 709;"RST"                   !Reset HP 3245A
30   OUTPUT 709;"SCRATCH"               !Clear HP 3245A memory
40   OUTPUT 709;"RQS 8"                 !Unmask bit 3 (LOCAL) in RQS mask
50   WHILE NOT BIT (SPOLL(709),6)       !Loop until Local key pressed
60   END WHILE                          !End loop
70   PRINT "HP 3245A NOW IN LOCAL"      !Display message
80   END
```

RSTATE

Description

Recall Stored State. States stored with **SSTATE** or **SSTATEn** can be recalled with the **RSTATE** or **RSTATEn** commands.

Syntax

RSTATE *number* or **RSTATEn** *number*

Parameters

number

Number of the stored state. For single-channel instrument operation, *number* = 0 through 13. For two-channel instrument operation, *number* = 0 through 6. Also, for single-channel instrument operation, only **RSTATE** and **RSTATEA** apply. For two-channel operation, **RSTATE**, **RSTATEA**, and **RSTATEB** apply.

Remarks

Setting/Recalling Channel States

The hardware state of a channel (such as DCV output, the arbitrary waveform contents, frequency = 1 kHz, DC offset = 2 VDC, etc.) can be stored in continuous (nonvolatile) memory with **SSTATE** or **SSTATEn** (**DRIVETBn** is stored by **SSTATE** only). The stored state of the channel can then be recalled (and the state of the channel restored) with the **RSTATE** or **RSTATEn** command (**DRIVETBn** is recalled by **RSTATE** only).

Since the states stored in continuous memory are not destroyed when power is removed, you can use a single **RSTATE** command to restore the channel to the state existing before power-down, rather than having to reenter all the commands required to set the channel parameters.

NOTE

If a power failure occurs while a state is being stored in continuous memory, that state becomes invalid. When power is restored, the invalid partial state is purged. However, states stored in continuous memory are not purged with a **SCRATCH** *command nor are stored states lost at power-down.*

Single-Channel Instrument Operation

For single-channel instrument operation, the current state of channel A is initially stored in volatile memory. Use **SSTATE** *number* or **SSTATEA** *number* to store the state of channel A into the state number specified by *number*. For example, **SSTATE 3** or **SSTATEA 3** stores the existing state of channel A into state #3.

For a single-channel instrument, you can store up to 14 states (0 through 13) in continuous memory using **SSTATE** or **SSTATEA** *number*. When a state is stored, use **RSTATE** *number* or **RSTATEA** *number* to recall the state from continuous to volatile memory and thus restore the channel state. For example, **RSTATE 2** or **RSTATEA 2** transfers the state stored in continuous memory state #2 into volatile memory.

Two-Channel Instrument Operation

With two-channel operation, seven states (0 through 6) are specified for each channel. That is, state 0 stores information in state A(0) and B(0), state 1 in A(1) and B(1), etc. When **SSTATE** *number* is specified, the state of channel A is stored in A(n) and the state of channel B in B(n).

If **SSTATEA** is used, only the state of channel A is stored. If **SSTATEB** is used, only the state of channel B is stored. For example, **SSTATE 1** stores the state of channels A and B in continuous memory in state 1 [A(1) and B(1)]. However, **SSTATEA 1** stores only the state of channel A in state 1 [A(1)].

Similarly, **RSTATE** *number* recalls the states of channels A and B from the specified continuous memory state, **RSTATEA** *number* recalls the state of channel A, and **RSTATEB** recalls the state of channel B.

RSTATE May Halt Waveform Output

Although the settings of a waveform with **TRIGMODE ARMWF** set (stored with **SSTATE** or **SSTATEn**) are recalled with **RSTATE** or **RSTATEn**, the waveform output will be suspended whether or not the waveform was running when stored. Therefore, to start a waveform when **TRIGMODE ARMWF** is set, a trigger must be sent to that channel (e.g., with **TRIGIN, PHSYNC**, or by applying an external signal).

RSTATE May Change Freq Ref Connector State

The driving channel and/or state of the Freq Ref connector may be changed by **STOREATOB**, **STOREBTOA**, or **RSTATE(A,B)**. If, for example, a 2-channel HP 3245A is in its power-on/reset state of channel A: **REFOUT EXT** and channel B: **REFOUT OFF, STOREBTOA** will set both channels to **REFOUT OFF** and they will no longer by synchronized.

Or, if channel A is set to **REFOUT EXT**, **STOREATOB** will set channel B to **REFOUT EXT** (which sets channel A to **REFOUT OFF**). A similar situation will result when TB0 or TB1 is driven with **SYNCOUT** or **REFOUT**.

Related Commands

SSTATE, SSTATEn, SET, SET?

Examples

Example: Recall Stored State (RSTATE)

This program sets channel A to output a 1.0 VDC signal, stores the channel A state (and the channel B state) in state #1, and pauses. After power is cycled and the program is continued, the channel A state (and the channel B state) are recalled with **RSTATE**. When the program pauses (line 90), cycle power and then press the controller Continue (or equivalent) key to complete the program.

```
10   !file RSTATE
20   !
30   CLEAR 709                    !Clear HP 3245A
40   OUTPUT 709;"RST"             !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"         !Clear HP 3245A memory
60   !
70   OUTPUT 709;"APPLY DCV 1.0"   !Output 1.0 VDC from ch A
80   OUTPUT 709;"SSTATE 1"        !Store ch state in state 1
90   PAUSE                        !Pause controller
100  OUTPUT 709;"RSTATE 1"        !Recall channel states
110  OUTPUT 709;"OUTPUT?"         !Query output value
120  ENTER 709;A                  !Enter value
130  PRINT A                      !Display value
140  END
```

Since **SSTATE 1** stored the state of channel A in continuous memory (in state 1), the state information is not lost when power is cycled. As a result, **RSTATE 1** sets the channel A output back to 1.0 VDC. Then, **OUTPUT?** returns "1.0" to verify that the channel A output is 1.0 VDC after power was cycled.

Description

Run Subroutine. Executes the named subroutine in parallel with other commands. Thus, subroutine commands are executed as the HP 3245A finds time between executing other commands (contrast with **CALL**).

Syntax

RUN *sub_name*

Parameters

sub_name Name of the subroutine which is run.

Remarks

Subroutine Must Be In Memory

The specified subroutine must be stored in HP 3245A memory using the **SUB** and **SUBEND** commands before executing the **RUN** command.

Subroutines and Commands Occur In Parallel

Subroutines executed with **RUN** proceed in parallel with commands from the front panel or from the controller. This parallel processing means that the state of the HP 3245A or subroutine variable values can be read while the subroutine is running. However, parallel execution does not allow two commands to occur at the same time.

Do Not Use RUN for Nesting Subroutines

The **RUN** command cannot be used to nest subroutines.

Execution Error Aborts Subroutine

If an error occurs during subroutine execution, the subroutine is aborted and the cause of the subroutine error is stored in the error register. Control of the HP 3245A is returned to the controller or to the front panel (wherever the **RUN** command initiated).

Using the USE Command Within Subroutines

A **USE** channel assignment within a subroutine executed by **RUN** is valid ONLY during subroutine execution. For example, suppose channel B is selected as the **USE** channel before the subroutine is run, but channel A is set as the **USE** channel in the subroutine. In this case, Channel A is used only in the subroutine and the **USE** channel assignment reverts back to channel B after the subroutine completes.

Related Commands

CALL, SUB, SUBEND, USE

Example

Example: Running a Subroutine (RUN)

This program initiates a RUN subroutine with a **RUN** command. It shows that a resource, such as the **USE** channel, specified within a RUN subroutine applies ONLY to that subroutine. When the program executes, channel A is designated as the **USE** channel. However, channel B (**USE 100**) is designated as the **USE** channel within the subroutine only. Thus, when the subroutine completes, **APPLY DCV 10** (line 150) changes the channel A output from 5.0 VDC to 10.0 VDC.

```
10   !file RUN
20   !
30   CLEAR 709                      !Clear HP 3245A
40   OUTPUT 709;"RST"               !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"           !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"             !Use channel A
80   OUTPUT 709;"  APPLY DCV 5"     !Output 5 VDC on channel A
90   OUTPUT 709;"  WAIT 5"          !Hold output value for 5 sec
100  OUTPUT 709;"SUB CHECK"         !Begin subroutine CHECK
110  OUTPUT 709;"  USE 100"         !Use channel B
120  OUTPUT 709;"    APPLY DCV 5"   !Output 5 VDC on channel B
130  OUTPUT 709;"SUBEND"            !End subroutine CHECK
140  OUTPUT 709;"RUN CHECK"         !Run subroutine CHECK
150  OUTPUT 709;"APPLY DCV 10"      !Output 10 VDC on channel A
160  END
```

Description

Running Query. Returns "1" if the current subroutine is running (or paused) or returns "0" otherwise. Only subroutines executed with the **RUN** command can be paused.

Syntax

RUNNING?

Parameters

None.

Remarks

Data Returned

RUNNING? returns "1" if the current subroutine is running (or paused) or "0" if the subroutine has finished running.

Data Destination

If **MEM** is used, the value returned by **RUNNING?** is stored in the specified variable or array. When **RUNNING?** is executed from the front panel, the response is displayed on the front panel display. When **RUNNING?** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

HP-IB Data Format

The HP 3245A returns numeric results in either ASCII or binary format (see the **OFORMAT** command). In the default ASCII format, **RUNNING?** returns integer results in 6-digit signed notation. In the binary format, **RUNNING?** returns integer results in 16-bit, 2's complement notation.

Related Commands

PAUSE, RUN, SUB, SUBEND

Example

Example: Reading Subroutine State (RUNNINGQ)

If this program is preceded by the program RUN (refer to the **RUN** command), executing this program returns the data shown below.

```
10   !file RUNNINGQ
20   !
30   OUTPUT 709;"RUNNING?"                     !Query state
40   ENTER 709;A                               !Enter state
50   PRINT "Subroutine State (1 = Running) =";A   !Display results
60   END
```

A typical return when the subroutine is finished is:

```
Subroutine State (1 = Running) = 0
```

SCRATCH

Description

Delete All. Deletes (scratches) all user-defined arrays, variables, and subroutines from HP 3245A volatile memory.

Syntax

SCRATCH

Parameters

Remarks

SCRATCH Versus DELSUB

The **DELSUB** (delete subroutine) command removes a specific subroutine but the subroutine name remains defined (i.e., the name cannot be used to define another variable, array, etc.). Unlike the **DELSUB** command, **SCRATCH** deletes arrays, variables, and subroutines from volatile memory and also removes the name definitions from the catalog (**CAT**) listing. If **SCRATCH** is executed when a subroutine is running, an error is generated and the subroutine is not purged from memory.

Related Commands

CAT, DELSUB

Example

Example: Clearing HP 3245A Memory (SCRATCH)

This program uses **SCRATCH CONT** to clear both volatile and continuous memory. The **CAT** command is used to show the operation of **SCRATCH** by returning a list of all user-defined arrays, variables, subroutines, and stored states.

```
10   !file SCRATCH
20   !
30   DIM Name$[60]                    !Dimension controller array
40   CLEAR 709                        !Clear HP 3245A
50   OUTPUT 709;"RST"                 !Reset HP 3245A
60   !
70   OUTPUT 709;"SUB BEEPER"          !Begin subroutine BEEPER
80   OUTPUT 709;"  BEEP"              !Beep once
90   OUTPUT 709;"SUBEND"              !End subroutine BEEPER
100  !
110  OUTPUT 709;"REAL R"              !Define REAL variable R
120  OUTPUT 709;"USE 0"               !Use channel A
130  OUTPUT 709;"  APPLY DCV 5"       !Output 5 VDC
140  OUTPUT 709;"INTEGER A(20)"       !Define A as INTEGER array
150  !
160 !OUTPUT 709;"SCRATCH"             !Delete all vars,arrays,subs
170  OUTPUT 709;"CAT"                 !Request CAT listing
180  !
190  REPEAT                           !Repeat loop
200    ENTER 709;Name$                !Enter line
210    PRINT Name$                    !Display line
220  UNTIL Name$="DONE"               !DONE is last word returned
230  END
```

When this program executes as listed (with line 160 as a comment line), a typical return is as follows.

```
IARRAY  A       SIZE    21
REAL    R
SUB     BEEPER  SIZE    88,  TEXT SIZE    62
DONE
```

If the exclamation point (!) is removed from line 160 and the program rerun, the data returned (DONE) shows that **SCRATCH** has deleted all arrays, variables, and subroutines from HP 3245A volatile memory. A typical return is:

```
DONE
```

SECURE

Description

Calibration Security. Sets a security code to prevent accidental or unauthorized calibration of the 3245A.

Information on the use of the SECURE command is found in the HP 3245A Calibration Manual (P/N 03245-90003).

Description

Serial Number Query. Returns "0000A00000".

Syntax

SER?

Parameters

None.

Remarks

Data Returned

On some Hewlett-Packard equipment, the **SER?** command returns the actual serial number of the instrument. However, the HP 3245A serial number cannot be read, so **SER?** returns "0000A00000".

Data Destination

If **MEM** is used, the value returned by **SER?** is stored in the specified variable or array. When **SER?** is executed from the front panel, the is displayed on the front panel display. When **SER?** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

Example

Example: Serial Number Query

```
10  OUTPUT 709;"SER?"     !Query serial number
20  ENTER 709;A$          !Enter value
30  PRINT A$              !Display  "0000A00000"
40  END
```

SET/SET?

Description

Send/Read HP 3245A State. **SET** instructs the HP 3245A to accept a binary block of data from the controller which specifies the HP 3245A configuration. Before executing **SET**, **SET?** must be executed to read the present HP 3245A configuration into the controller.

Syntax

SET *block_data*

Parameters

block_data

The *block_data* parameter is a block in IEEE-728 Block A format which specifies the configuration. The contents of the block must be identical to the block returned by the **SET?** command.

Remarks

SET? Command Operation

The **SET?** command transfers the configuration of the HP 3245A into the controller. The configuration can then be reloaded into the HP 3245A (with **SET**) should the instrument lose power or the configuration can be loaded into another HP 3245A, as required.

Block A Format

The *block_data* parameter is in IEEE-728 Block A format. Block A format header consists of four bytes: the # sign, the letter A, and two bytes which indicate the number of bytes of data to follow. When the HP 3245A configuration is transferred to the controller (with **SET?**) and is then returned to the HP 3245A (with **SET**), the Block A format is automatically provided.

Related Commands

BLOCKOUT, OFORMAT

Example

Example: Sending/Reading HP 3245A State (SET)

This program reads the configuration of the HP 3245A into the controller with the **SET?** command. The HP 3245A configuration is then sent to the HP 3245A from the controller using the **SET** command.

When the program executes, the state of the HP 3245A is transferred to the controller and stored in Set$ and the program is paused. When the Continue key is pressed, the program resumes, resets the HP 3245A, and then restores the previous state to the instrument with the **SET** command.

```
10   !file SET
20   !
30   DIM Set[8500]                          !Dim cont array
40   CLEAR 709                              !Clear HP 3245A
50   OUTPUT 709;"RST"                       !Reset HP 3245A
60   OUTPUT 709;"SCRATCH"                   !Clear HP 3245A memory
70   OUTPUT 709;"APPLY DCV 1.5"             !Output 1.5 VDC
80   !
90   OUTPUT 709;"SET?"                      !Read HP 3245A state
```

```
100  ENTER 709;Set$                    !Enter state
110  DISP "PRESS CONTINUE TO RESUME"    !Display message
120  PAUSE                              !Pause program
130  DISP ""                            !Clear display
140  OUTPUT 709;"RST"                   !Reset HP 3245A
150  OUTPUT 709;"SET";Set$              !Trans state to HP 3245A
160  END
```

SET TIME

Description

Set Time. Sets the HP 3245A internal clock in number of seconds since midnight. The time is stored in volatile memory and is lost when power is removed.

Syntax

SET TIME *seconds*

Parameters

seconds Number of seconds since midnight. The valid range is 0 through 86399.9 (resolution = 0.01 seconds).

Remarks

Power-On/Reset Value of Internal Clock

At power-on/reset, the HP 3245A internal clock is set to 0.0 seconds. The time set by **SET TIME** is stored in volatile memory and is lost when power is removed.

Converting Time Formats

HP 9000 Series 200/300 controllers use two commands to convert time formats: **TIME** and **TIME$**. The **TIME** command converts a formatted time-of-day string (hh:mm:ss) into a numeric value of seconds since midnight. The **TIME$** command converts the number of seconds value into a string representing the time-of-day format.

You can calculate a specific time in seconds since midnight by using t (seconds) = (hours*3600) + (minutes*60) + seconds. For example, 9:45:30 A.M. = (9 x 3600) + (45 x 60) + 30 = 35130 seconds. Or, for 9:45:30 A.M., the number of seconds since midnight can be specified as: `OUTPUT 709;"SET TIME";(9*3600)+(45*60)+30`.

Related Commands

TIME

Example

Example: Setting Clock Time (SET_TIME)

This program sets the HP 3245A internal clock to 9:55 A.M. and verifies that the clock is running by reading the time five seconds after the clock is set. The time is set using the HP 9000 Series 200/300 TIME command and the return time (09:55:05) is converted to an hh:mm:ss format by the TIME$ command.

```
10   !file SET_TIME
20   !
30   CLEAR 709                 !Clear HP 3245A
40   OUTPUT 709;"RST"          !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"      !Clear HP 3245A memory
60   !
70   OUTPUT 709;"SET TIME";TIME ("9:55:00")   !Set time to 9:55 A.M.
80   WAIT 5                    !Wait five seconds
90   OUTPUT 709;"TIME"         !Read clock time
100  ENTER 709;A               !Enter time
110  PRINT TIME$(A)            !Convert time and display
120  END
```

Description

Shift Bits Without Wraparound. Returns an integer obtained by shifting the argument a specified number of positions without bit wraparound (contrast with **ROTATE**).

Syntax

SHIFT (*argument*, *bit_displacement*)

Parameters

argument

Must be a numeric integer within the range of -32768 to +32767. Within the HP 3245A, the argument is represented as a 16-bit, 2's complement integer.

bit_displacement

Specifies the number of positions bits are shifted. Positive values shift the argument toward the least significant bit and negative values shift the argument toward the most significant bit.

Remarks

Related Commands

ROTATE

Examples

Example: Positive Shift

This program rotates the bit pattern for 12 three positions toward the least significant bit as shown and displays the result (1) on the controller CRT.

```
      12 = 0000 0000 0000 1100
Shift #1 = 0000 0000 0000 0110
Shift #2 = 0000 0000 0000 0011
Shift #3 = 0000 0000 0000 0001
```

```
10  OUTPUT 709;"VREAD SHIFT(12,3)"    !Shift bit pattern for 12
20  ENTER 709;A                       !Enter result
30  PRINT "Shift Result = ";A         !Display result
40  END
```

Example: Negative Shift

This program shifts the bit pattern for 12 three positions toward the most significant bit as shown and displays the result (96) on the controller CRT.

```
      12 = 0000 0000 0000 1100
Shift #1 = 0000 0000 0001 1000
Shift #2 = 0000 0000 0011 0000
Shift #3 = 0000 0000 0110 0000
```

```
10  OUTPUT 709;"VREAD SHIFT(12,-3)      !Shift the bit pattern for 12
20  ENTER 709;A                         !Enter result
30  PRINT "Shift Result = ";A           !Display result
40  END
```

SIN

Description

Sine. Returns the sine of the angle (in radians) represented by the argument.

Syntax

SIN (*argument*)

Parameters

argument

Must be a number or numeric expression (enclosed in parentheses) in radians and in the range ± 2.98156826 E+8.

Remarks

Related Commands

ATN, COS

Example

Example: Using SIN Function

This program computes the sine of 0.5235988 radians (30 degrees) and displays the result (.5) on the controller CRT.

```
10  OUTPUT 709;"VREAD SIN(.5235988)"   !Compute sine of 30 degrees
20  ENTER 709;A                        !Enter result
30  PRINT "Sine = ";A                  !Display result
40  END
```

A typical return is:

```
    Sine = .50000002
```

Description

Size Query. Returns the number of elements in the specified array.

Syntax

SIZE? *array_name*

Parameters

array_name Array whose size is returned.

Remarks

Data Returned

SIZE? returns the number of elements in the specified array. This number is one more than the index of the last element in the array due to the option base 0 convention (see **DIM, REAL**, or **INTEGER** commands). If the number of the maximum element is 32767, **SIZE?** returns -32768 since +32768 cannot be represented as a 16-bit signed integer.

NOTE

For REAL arrays, ERROR 41 - OUT OF MEMORY is generated if you try to define the array size greater than about 10000 elements.

Data Destination

If the **MEM** command is used, the value returned by **SIZE?** is stored in the specified variable or array. When **SIZE?** is executed from the front panel, the response is displayed on the front panel display. When **SIZE?** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

HP-IB Data Format

The HP 3245A returns numeric results in either ASCII or binary format (see the **OFORMAT** command). In the default ASCII format, **SIZE?** returns integer results in 6-digit signed notation. In the binary format, **SIZE?** returns integer results in 16-bit, 2's complement notation.

Related Commands

DIM, INTEGER, REAL

Example

Example: Using the SIZE? Command

This program defines INTEGER array VOLT1 with 10 elements and then verifies the array size using the **SIZE?** command.

```
10  OUTPUT 709;"INTEGER VOLT1(9)"   !Integer array with 10 elements
20  OUTPUT 709;"SIZE? VOLT1"        !Read array size
30  ENTER 709;A                     !Return array size
40  PRINT "VOLT1 Size = ";A         !Display array size
50  END
```

SPOLL (Serial Poll)

Description

SPOLL, like the HP 3245A **STB?** command, returns the status byte which is the weighted sum of the lower eight bits set in the Status Register.

Syntax

P=SPOLL (709)

Remarks

- **SPOLL** returns a decimal value which is the sum of lower eight bits in the HP 3245A Status Register bits which are true (set). A "0" is returned if no Status Register bits are set. The following table defines Status Register bits and associated decimal values.

Bit	Value	Description
0	1	DATA AVAILABLE. Set (1) when data is available in the output buffer. Cleared (0) when data is removed from the buffer; by the CLROUT, CLR, or CLEAR commands; or with the front panel Clear key.
1	2	NOT USED.
2	4	USER SERVICE REQUEST. Set (1) when SRQ is executed over HP-IB or from the front panel. Cleared (0) with STA?, CLR, CLEAR, or front panel Clear key.
3	8	LOCAL. Set (1) when the HP 3245A powers on; when it is reset (RST); or when it enters the Local mode. Cleared (0) with STA?, CLR, or CLEAR.
4	16	READY. Set (1) when the input buffer is empty and the HP 3245A is not executing a command or a CALL-executed subroutine. Cleared (0) when the HP 3245A is executing a command or a CALL-executed subroutine.
5	32	ERROR. Set true (1) when an error condition occurs. Cleared (0) when ERR? or ERRSTR? clear the error register or by CLR or CLEAR.
6	64	SRQ SENT. Set (1) (and HP-IB SRQ sent to controller) when any other unmasked bit (bit 0, 2, 3, 4, or 5) in the status register is set. Cleared (0) by STA?, CLR, CLEAR, SPOLL or by the front panel Clear key.
7-16	---	NOT USED.

- If the HP-IB SRQ line is set TRUE when **SPOLL** is sent, all bits in the Status Register are cleared if the condition which set the bit(s) is no longer present. If the SRQ line is FALSE when **SPOLL** is sent, the Status Register contents are not changed.

- **SPOLL** differs from **STB?** in that **STB?** interrupts the HP 3245A microprocessor. With **STB?**, the HP 3245A always appears to be busy (bit 4 clear). However, **SPOLL** reads the status byte without interrupting the microprocessor, so **SPOLL** can be used to monitor the ready state of the HP 3245A.

- If there is data in the output buffer when **SPOLL** is sent, the data remains intact. However, if there is data in the output buffer when **STB?** is sent, that data is overwritten by the status data (unless **OUTBUF ON** is set).

Example

```
10  P=SPOLL (709)      !Send Serial Poll, place response in P
20  DISP P             !Display response
30  END
```

SQR

Description

Square Root. Returns the square root of the specified argument.

Syntax

SQR (*argument*)

Parameters

argument — Must be a number or numeric expression (enclosed in parentheses) greater than or equal to zero.

Remarks

None.

Example

Example: Square Root of a Number

This program computes the square root of 2.345 and displays the result (1.5313393) on the controller CRT.

```
10  OUTPUT 709;"VREAD SQR(2.345)"   !Compute square root
20  ENTER 709;A                     !Enter result
30  PRINT "Square Root = ";A        !Display result
40  END
```

A typical return is:

```
    Square Root = 1.5313393
```

Description

Programmed Service Request. If status register bit 2 (USER SERVICE REQUEST) is unmasked by the **RQS** command, executing **SRQ** sets the HP-IB SRQ line TRUE which signals the controller that the HP 3245A has requested service.

Syntax

SRQ

Parameters

None.

Remarks

Service Request Action

When executed, **SRQ** sets bit 2 in the status register. If the USER SERVICE REQUEST bit (bit 2) in the RQS mask register is unmasked (with **RQS 4**), the HP-IB SRQ line is asserted and the controller is notified of the HP 3245A service request.

Using Front Panel SRQ

Entering the **SRQ** command via the front panel or from an HP 3245A subroutine performs the same action as entering the **SRQ** command via HP-IB from the controller.

Related Commands

PONSRQ, RQS, RQS?, STA?

Example

Example: Using Front Panel SRQ (SRQ)

This program shows one way to use the **SRQ** command from the front panel to generate an interrupt signal to the controller. The program loops continuously until an **SRQ** command is executed from the front panel (via the MENU keys). When **SRQ** is executed, the front panel SRQ annunciator is lit and a message is displayed on the controller CRT.

```
10   !file SRQ
20   !
30   CLEAR 709                         !Clear HP 3245A
40   OUTPUT 709;"RST"                  !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"              !Clear HP 3245A memory
60   OUTPUT 709;"RQS 4"                !Unmask bit 2 (USER SERV REQ)
70   !
80   ON INTR 7 GOTO 120                !Go to line 120 when SRQ exec
90   ENABLE INTR 7;2                   !Enable HP-IB SRQ interrupt
100  WHILE 1                           !Loop until SRQ executed
110  END WHILE                         !End loop
120  PRINT "3245 SRQ RECEIVED"         !Display interrupt message
130  OUTPUT 709;"STA?"                 !Read status register bits set
140  ENTER 709;A                       !Enter bits set
150  PRINT                             !Space
160  PRINT "Status Reg. Bits Set = ";A !Display bits set
170  END
```

When **SRQ** is executed from the front panel, the controller CRT display is as follows. The returned value of 68 shows that bit 6 - SRQ SENT (weight = 64) and bit 2 - USER SERVICE REQUEST (weight = 4) are set (68 = 64 + 4).

```
3245 SRQ RECEIVED

Status Reg. Bits Set = 68
```

Description

Store HP 3245A State. Stores up to 14 HP 3245A states in continuous memory. Stored states can be recalled with the **RSTATE** or **RSTATEn** commands.

Syntax

SSTATE *number* or **SSTATEn** *number*

Parameters

number

Number of the stored state. For single-channel instrument operation, *number* = 0 through 13. For two-channel instrument operation, *number* = 0 through 6. Also, for single-channel instrument operation, only **SSTATE** and **SSTATEA** apply. For two-channel operation, **SSTATE**, **SSTATEA**, and **SSTATEB** apply.

Remarks

Setting/Recalling Channel States

The hardware state of a channel (such as DCV output, the arbitrary waveform contents, frequency = 1 kHz, DC offset = 2 VDC, etc.) can be stored in continuous (nonvolatile) memory with **SSTATE** or **SSTATEn** (**DRIVETBn** is stored by **SSTATE** only). The stored state of the channel can then be recalled (and the state of the channel restored) with the **RSTATE** or **RSTATEn** command (**DRIVETBn** is recalled by **RSTATE** only).

Since the states stored in continuous memory are not destroyed when power is removed, you can do a single **RSTATE** command to restore the channel to the state existing before power-down, rather than having to re-enter all the commands required to set the channel parameters.

NOTE

If a power failure occurs while a state is being stored in continuous memory, that state becomes invalid. When power is restored, the invalid partial state is purged. However, states stored in continuous memory are not purged with a **SCRATCH** *command nor are stored states lost at power-down.*

Single-Channel Instrument Operation

For single-channel instrument operation, the current state of channel A is initially stored in volatile memory. Use **SSTATE** *number* or **SSTATEA** *number* to store the state of channel A into the state number specified by *number*. For example, **SSTATE 3** or **SSTATEA 3** stores the existing state of channel A into state #3.

For a single-channel instrument, you can store up to 14 states (0 through 13) in continuous memory using **SSTATE** or **SSTATEA** *number*. When a state is stored, use **RSTATE** *number* or **RSTATEA** *number* to recall the state from continuous to volatile memory and thus restore the channel state. For example, **RSTATE 2** or **RSTATEA 2** transfers the state stored in continuous memory state #2 into volatile memory.

Two-Channel Instrument Operation

With two-channel operation, seven states (0 through 6) are specified for each channel. That is, state 0 stores information in state A(0) and B(0), state 1 in A(1) and B(1), etc. When **SSTATE** *number* is specified, the state of channel A is stored in A(n) and the state of channel B in B(n).

If **SSTATEA** is used, only the state of channel A is stored. If **SSTATEB** is used, only the state of channel B is stored. For example, **SSTATE 1** stores the state of channels A and B in continuous memory in state 1 [A(1) and B(1)]. However, **SSTATEA 1** stores only the state of channel A in state 1 [A(1)].

Similarly, **RSTATE** *number* recalls the states of channels A and B from the specified continuous memory state, **RSTATEA** *number* recalls the state of channel A, and **RSTATEB** recalls the state of channel B.

Using STOREATOB/STOREBTOA for Two-Channel Operation

In addition, for two-channel instrument operation, you can use **STOREATOB** to copy the state of channel A into channel B or use **STOREBTOA** to copy the state of channel B into channel A (both previous states are still stored in continuous memory). Then, you can use **SSTATEA** or **SSTATEB** to transfer the state to continuous memory.

Related Commands

RSTATE, RSTATEn, SET, SET?

Examples

Example: Store State (SSTATE)

This program sets channel A to output a 1.0 VDC signal, stores the channel A state (and the channel B state) in state #1, and pauses. After power is cycled and the program is continued, the channel A state (and the channel B state) are recalled with **RSTATE**. When the program pauses (line 80), cycle power and then press the controller Continue (or equivalent) key to complete the program.

```
10   !file SSTATE
20   !
30   CLEAR 709                     !Clear HP 3245A
40   OUTPUT 709;"RST"              !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"          !Clear HP 3245A memory
60   OUTPUT 709;"APPLY DCV 1.0"    !Output 1.0 VDC from ch A
70   OUTPUT 709;"SSTATE 1"         !Store ch state in state 1
80   PAUSE                         !Pause controller
90   OUTPUT 709;"RSTATE 1"         !Recall channel states
100  END
```

Description

Status Word Query. Returns the status word. The number returned by **STA?** is the weighted sum of all bits (masked or unmasked) set in the status register.

Syntax

STA?

Parameters

None.

Remarks

Data Returned

STA? returns a decimal value which is the sum of the status register bits which are TRUE (set). A "0" is returned if no status register bits are set.

Power-On Conditions

At power-on, **STA?** returns "0" if power-on SRQ is disabled (**PONSRQ OFF**). If power-on SRQ is enabled (**PONSRQ ON**), at power-on **STA?** returns "72" and the HP-IB SRQ line is asserted.

Reading/Clearing Status Register with STA?

The **STA?** command reads the status register bits set and clears certain bits in the register after the read has been made. **STA?** returns a decimal value equal to the sum of the decimal values of all status register bits set (both masked and unmasked). For example, if bit 0 - DATA AVAILABLE (value = 1) and bit 4 - READY (value = 16) are set, **STA?** returns 17.

STA? also clears status register bits after the read has been made. **STA?** clears bit 2 - USER SERVICE REQUEST and bit 3 - LOCAL after the command executes. For example, if both of these bits are unmasked and both bits are set, status register bit 6 - SRQ SENT is also set and an SRQ is sent to the controller. In this case, **STA?** returns 76 (64 + 8 + 4) and clears bits 2, 3, and 6.

Executing **STA?** clears bits 2 and 3 which are set by an event (USER SERVICE REQUEST and LOCAL). Bits 4 and 5 which reflect state (condition) (DATA AVAILABLE, READY, and ERROR) are not cleared by **STA?**. Bit 6 (SRQ SENT) and the HP-IB SRQ line are cleared ONLY if no unmasked bits remain set.

NOTE

When **STA?** *or* **STB?** *is executed, bit 4 - READY, will always show cleared since the HP 3245A is busy executing the command. Use* **READY?** *or* **SPOLL** *to check the ready status of the HP 3245A.*

Data Destination

If **MEM** is used, the value returned by **STA?** is stored in the specified variable or array. When **STA?** is executed from the front panel, the response is displayed on the front panel display. When **STA?** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

Status Register Bit Definitions

This table defines status register bits and associated decimal values. It also shows the events/conditions which set/clear the bits.

Bit	Value	Definition	Set by:	Cleared by:
0	1	DATA AVAILABLE	Data available in the output buffer.	CLR, CLEAR, CLROUT, or Clear key or when data is removed from output buffer
1	2	NOT USED		
2	4	USER SERVICE REQUEST	SRQ command from controller or from HP 3245A.	STA?, CLR, CLEAR, or Clear key.
3	8	LOCAL	Power-on, RST, or Local mode set.	STA?, CLR, CLEAR, or Clear key.
4	16	READY	Input buffer is empty and no command or CALL subroutine is executing.	Command or CALL sub-routine is executing.
5	32	ERROR	An error condition occurring.	CLR, CLEAR or when ERR? or ERRSTR? clears error register.
6	64	SRQ SENT	When any unmasked bit (0, 2, 3, 4, or 5) in status register is set.	STA?, STB?, CLR, CLEAR, SPOLL, or Clear key.*
7-16		NOT USED		

* = Not cleared by STA?/STB? for every condition.

HP-IB Data Format

The HP 3245A returns numeric results in either ASCII or binary format (see the **OFORMAT** command). In the default ASCII format, **STA?** returns integer results in 6-digit signed notation. In the binary format, **STA?** returns integer results in 16-bit, 2's complement notation.

Related Commands

PONSRQ, RQS, STB?

Example

Example: Reading the Status Word (STAQ)

This program uses **STA?** to read the system status word. The value returned is the weighted sum of the status register bits (masked or unmasked) which are set. In the program, **IMP?** returns data to the output buffer, which sets status register bit 0 - DATA AVAILABLE. Then, **STA?** returns a "1" (decimal value of bit 0) to indicate bit 0 is set. (Note that line 80 is a comment line. It is added so that this program can be modified and used for **STB?**.)

```
10   !file STAQ
20   !
30   CLEAR 709                          !Clear HP 3245A
40   OUTPUT 709;"RST"                   !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"               !Clear HP 3245A memory
60   OUTPUT 709;"IMP?"                  !Query impedance
70   OUTPUT 709;"STA?"                  !Read status word
80  !OUTPUT 709;"STB?"                  !Read status byte
90   ENTER 709;A                        !Enter value
100  PRINT                              !Space
110  PRINT "Status Reg Bits Set = ";A   !Display value
120  END
```

Since bit 0 is the only status register bit set, **STA?** returns the weighted decimal value (1) of the bits set and a typical return is:

```
    Status Reg Bits Set = 1
```

STB?

Description

Status Byte Query. Returns the status byte. The number returned by **STB?** is the weighted sum of all bits (masked or unmasked) set in the status register.

Syntax

STB?

Parameters

None.

Remarks

Data Returned

STB? returns a decimal value which is the sum of the status register bits which are TRUE (set). A "0" is returned if no status register bits are set.

Power-On Conditions

At power-on, **STB?** returns "0" if power-on SRQ is disabled (**PONSRQ OFF**). If power-on SRQ is enabled (**PONSRQ ON**), at power-on **STB?** returns "72" and the HP-IB SRQ line is asserted.

Reading/Clearing Status Register with STB?/SPOLL

STA?, **STB?**, and **SPOLL** all return the weighted sum of the status register bit which are set (both masked and unmasked). However, **STA?** also clears bit 2 (USER SERVICE REQUEST) and bit 3 (LOCAL). If desired, you can use **STB?** to read the status register bits without clearing bits 0, 2, 3, 4, or 5.

Since **STB?** interrupts the HP 3245A microprocessor, when **STB?** is used the HP 3245A appears to be busy (bit 4 is cleared). Since **SPOLL** reads the status register without interrupting the microprocessor, you can use **SPOLL** to monitor the READY state of the HP 3245A.

If the HP-IB SRQ line is set TRUE when **SPOLL** is sent, all bits in the status register are cleared if the event or condition which set the bit(s) is no longer present. If the SRQ line is FALSE when **SPOLL** is sent, the status register contents are not changed.

If there is data in the output buffer when **SPOLL** is sent, the data remains intact. In contrast, if there is data in the output buffer when **STB?** is sent, that data is overwritten by the status data (unless **OUTBUF ON** is set).

NOTE

When **STA?** *or* **STB?** *is executed, bit 4 - READY, will always show cleared since the HP 3245A is busy executing the command. Use* **READY?** *or* **SPOLL** *to check the ready status of the HP 3245A.*

Data Destination

If **MEM** is used, the value returned by **STB?** is stored in the specified variable or array. When **STB?** is executed from the front panel, the response is displayed on the front panel display. When **STB?** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

Status Register Bit Definitions

This table defines the status register bits and associated decimal values. It also shows the events/conditions which set/clear the bits.

Bit	Value	Definition	Set by:	Cleared by:
0	1	DATA AVAILABLE	Data available in the output buffer.	CLR, CLEAR, CLROUT, or Clear key or when data is removed from output buffer
1	2	NOT USED		
2	4	USER SERVICE REQUEST	SRQ command from controller or from HP 3245A.	STA?, CLR, CLEAR, or Clear key.
3	8	LOCAL	Power-on, RST, or Local mode set.	STA?, CLR, CLEAR, or Clear key.
4	16	READY	Input buffer is empty and no command or CALL subroutine is executing.	Command or CALL sub-routine is executing.
5	32	ERROR	An error condition occurring.	CLR, CLEAR or when ERR? or ERRSTR? clears error register.
6	64	SRQ SENT	When any unmasked bit (0, 2, 3, 4, or 5) in status register is set.	STA?, STB?, CLR, CLEAR, SPOLL, or Clear key.*
7-16		NOT USED		

* = Not cleared by STA?/STB? for every condition.

HP-IB Data Format

The HP 3245A returns numeric results in either ASCII or binary format (see the **OFORMAT** command). In the default ASCII format, **STB?** returns integer results in 6-digit signed notation. In the binary format, **STB?** returns integer results in 16-bit, 2's complement notation.

Related Commands

PONSRQ, RQS, STA?

Example

Example: Reading the Status Byte

This program uses **STB?** to read the system status byte. The value returned is the weighted sum of the status register bits (masked or unmasked) which are set. In the program, **IMP?** returns data to the output buffer, which sets status register bit 0 - DATA AVAILABLE. Then, **STB?** returns a "1" (decimal value of bit 0) to indicate bit 0 is set. (If desired, you can use the STAQ program stored on the Example Programs disc and modify lines 70 and 80.)

```
10    !file STAQ
20    !
30    CLEAR 709                         !Clear HP 3245A
40    OUTPUT 709;"RST"                  !Reset HP 3245A
50    OUTPUT 709;"SCRATCH"              !Clear HP 3245A memory
60    OUTPUT 709;"IMP?"                 !Query impedance
70   !OUTPUT 709;"STA?"                 !Read status word
80    OUTPUT 709;"STB?"                 !Read status byte
90    ENTER 709;A                       !Enter value
100   PRINT                             !Space
110   PRINT "Status Reg Bits Set = ";A  !Display value
120   END
```

Since bit 0 is the only status register bit set, **STB?** returns the weighted decimal value (1) of the bits set and a typical return is:

```
    Status Reg Bits Set = 1
```

Description

Single Step Subroutine. Steps through the specified subroutine, line by line, to verify its operation.

Syntax

STEP [*sub_name*]

Parameters

sub_name The name of a previously stored subroutine.

Remarks

Using the STEP Command

To start the single-step process, send the **STEP** command followed by the name of the subroutine. Next, send the **STEP** command to sequence through the subroutine, line by line. Each time **STEP** is executed, the next subroutine line is displayed and executed. If the line generates data, the data is displayed.

You can execute the **CONT** (continue) command to cease stepping and continue normal execution of the subroutine from the last line stepped. Compressed subroutines (**COMPRESS** command) cannot be stepped.

Related Commands

COMPRESS, CONT, PAUSE, SUB, SUBEND

Example

Example: Stepping Through a Stored Subroutine

This program line steps through stored subroutine TEST1 one line at a time, each time line 200 is executed.

```
        .
        .
100  OUTPUT 709;"STEP TEST1"     !Step through subroutine TEST1
        .
        .
200  OUTPUT 709;"STEP"           !Step to next line
```

STOREATOB/STOREBTOA

Description

Copy Channel State. **STOREATOB** stores (copies) the present state of channel A into channel B. **STOREBTOA** stores the present state of channel B into channel A.

Syntax

STOREATOB or **STOREBTOA**

Parameters

None.

Remarks

Using STOREATOB/STOREBTOA

For two-channel instrument operation, use **STOREATOB** to copy the state of channel A into channel B or use **STOREBTOA** to copy the state of channel B into channel A (both previous states are still stored in continuous memory). After **STOREATOB** or **STOREBTOA** is executed, the states remain in volatile memory. Use **SSTATEA** or **SSTATEB** to transfer the state to continuous memory.

Changing Freq Ref Connector State

The driving channel and/or state of the Freq Ref connector may be changed by **STOREATOB**, **STOREBTOA**, or **RSTATE(A,B)**. If, for example, a 2-channel HP 3245A is in its power-on/reset state of channel A: **REFOUT EXT** and channel B: **REFOUT OFF**, **STOREBTOA** will set both channels to **REFOUT OFF** and they will no longer by synchronized. Or, if channel A is set to **REFOUT EXT**, **STOREATOB** will set channel B to **REFOUT EXT** (which sets channel A to **REFOUT OFF**). A similar situation will result when TB0 or TB1 is driven with **SYNCOUT** or **REFOUT**.

Related Commands

SSTATEA, SSTATEB

Example

Copying Channel A State to Channel B (STOREATOB)

This program sets channel A to a specified state (2.5 VDC offset, 10 kHz frequency, and 5 V ac PP square wave output) and then pauses. When you press the Continue key, the program continues and uses **STOREATOB** to copy the channel A state to channel B.

```
10   !file STOREATOB
20   !
30   CLEAR 709                      !Clear HP 3245A
40   OUTPUT 709;"RST"               !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"           !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"             !Use channel A
80   OUTPUT 709;"  DCOFF 2.5"       !2.5 VDC offset
90   OUTPUT 709;"  FREQ 10E3"       !10 kHz frequency
100  OUTPUT 709;"  APPLY SQV 5"     !5 V ac PP square wave
110  PAUSE                          !Pause controller
120  OUTPUT 709;"STOREATOB"         !Copy ch A state to ch B
130  END
```

Description

Begin Subroutine. Instructs the HP 3245A to store all subsequent commands until the **SUBEND** command in the named subroutine.

Syntax

SUB *sub_name*

Parameters

sub_name

Subroutine name. Subroutine names may contain up to 10 characters. The first character must be a letter (A-Z) but the remaining nine characters can be letters, numbers (0-9), the underscore character ("_"), or the question mark ("?"). Subroutine names must not be the same as HP 3245A commands or parameters, previously defined array or variable names, or stored state names.

Remarks

Do Not Enter Subroutines From Front Panel

Since the front panel has no features for editing subroutines, it is recommended that you first create subroutines on your controller and then download them to the HP 3245A.

Executing Subroutines

Subroutines can be executed using either the **CALL** command or the **RUN** command. See the **CALL** or **RUN** command for details.

Subroutine Nesting

Nesting subroutines allows one subroutine to execute (**CALL** only) another subroutine. Subroutines can be nested up to 10 deep.

Subroutine Syntax is Checked When Downloaded

When a subroutine is entered, the HP 3245A checks the subroutine for syntax errors. If the syntax is not correct, an error is generated and the command is not stored in the subroutine. In this case, edit the subroutine in the controller and download it again.

Maximum Number of Subroutines Allowed

The exact number of subroutines that can be stored depends upon the sizes of the individual subroutines. A subroutine which consists of about 10 commands (including **SUB** and **SUBEND**) might average about 600 bytes.

Related Commands

CALL, COMPRESS, DELSUB, FILLBIN, LIST, RUN, SUBEND

Example

Example: Creating a Subroutine

This program creates subroutine OUT1 which outputs 0.1 VDC from channel A and then beeps once when the subroutine is executed (with **CALL**).

```
10   CLEAR 709                      !Clear HP 3245A
20   OUTPUT 709;"RST"               !Reset HP 3245A
30   OUTPUT 709;"SCRATCH"           !Clear HP 3245A memory
40   !
50   OUTPUT 709;"SUB OUT1"          !Begin sub OUT1
60   OUTPUT 709;"  APPLY DCV .1"    !Output 0.1 VDC
70   OUTPUT 709;"  BEEP"            !BEEP once
80   OUTPUT 709;"SUBEND"            !End subroutine
90   OUTPUT 709; "CALL OUT1"        !Call subroutine
100  END
```

Description

End Subroutine. Identifies where the subroutine ends and also terminates the subroutine entry. The **SUBEND** command must be the last statement in the subroutine. All commands listed between **SUB** and **SUBEND** become part of the subroutine.

Syntax

SUBEND

Parameters

None.

Remarks

Subroutine Operations

Since the front panel has no features for editing subroutines, it is recommended that you first create subroutines on your controller and then download them to the HP 3245A. Subroutines may be executed using either **CALL** or **RUN**. Nesting subroutines allows one subroutine to execute (**CALL** only) another subroutine. Subroutines can be nested up to 10 deep.

Subroutine Syntax Checked When Downloaded

When a subroutine is entered, the HP 3245A checks the subroutine for syntax errors. If the syntax is not correct, an error is generated and the command is not stored in the subroutine. In this case, edit the subroutine in the controller and download it again.

Related Commands

CALL, COMPRESS, DELSUB, FILLBIN, LIST, RETURN, RUN, SUB

Example

Example: Creating a Subroutine

This program creates subroutine OUT1 which outputs 0.1 VDC from channel A and then beeps once when the subroutine is executed (with **CALL**).

```
10   CLEAR 709                   !Clear HP 3245A
20   OUTPUT 709;"RST"            !Reset HP 3245A
30   OUTPUT 709;"SCRATCH"        !Clear HP 3245A memory
40   !
50   OUTPUT 709;"SUB OUT1"       !Begin sub OUT1
60   OUTPUT 709;"  APPLY DCV .1" !Output 0.1 VDC
70   OUTPUT 709;"  BEEP"         !BEEP once
80   OUTPUT 709;"SUBEND"         !End subroutine
90   OUTPUT 709; "CALL OUT1"     !Call subroutine
100  END
```

SYNCOUT/SYNCOUT?

Description

Set/Read Sync Out. SYNCOUT controls the destination of the SYNC output signal for AC waveforms. **SYNCOUT?** returns the SYNC signal output destination (front panel Sync Out connector and/or rear panel TB0 or TB1 connector).

Syntax

SYNCOUT ***destination***

SYNCOUT?

Parameters

destination

The SYNC signal is always routed to the front panel Sync Out connector, but can also be routed to the TB0 or TB1 output connectors on the rear panel with the *destination* parameter as shown. Power-on/reset *destination* = OFF.

destination	Definition
OFF	Disable external routing of Sync signal (the Sync signal is routed only to the Sync Out connector).
TB0	Route Sync signal to Sync Out connector and to the TB0 connector on the rear panel.
TB1	Route Sync signal to Sync Out connector and to the TB1 connector on the rear panel.

Remarks

The Sync Out Signal

All AC waveforms (sine, ramp, square, and arbitrary) have an associated SYNC signal which is output from the front panel Sync Out connector when the waveform is output. The Sync signal is a logic "1" (+5V) when the waveform output level is positive relative to its DC offset. The Sync signal is a logic "0" (0V) when the waveform output level is negative relative to its DC offset.

Sync Out Signal Destinations

The Sync Out signal is always routed to the front panel Sync Out connector for the specified **USE** channel, but can also be routed to the TB0 or TB1 ports on the rear panel. **SYNCOUT OFF** disables external routing of sync signal and the signal is routed only to the Sync Out connector.

SYNCOUT TB0 or **TB1** automatically disables the software drivers (**DRIVETBn**) and/or the other channel's drive (**SYNCOUT** or **REFOUT**) to the selected trigger bus and routes the present **USE** channel's sync signal to the TB0 or TB1 port.

REFOUT/SYNCOUT/DRIVETBn Interaction

Using **REFOUT TBn** or **SYNCOUT TBn** automatically resets **DRIVETBn** to **OFF**. Using **DRIVETBn** other than **OFF**, **SYNCOUT TBn**, or **REFOUT TBn** from the other channel automatically resets **REFOUT TBn** on this channel to **REFOUT OFF**.

Query Command (SYNCOUT?)

The **SYNCOUT?** command returns the destination (front panel Sync Out connector and/or rear panel TB0 or TB1 connector) for the SYNC output signal.

Related Commands

DCOFF, USE

Example

Example: Using the Sync Signal (SYNCOUT)

This program uses the Sync Out signal to gate the channel B output. Channel A is set for a 1 kHz square wave @ 1 Vac PP and channel B is set for a 10 kHz sine wave @ 2.5 V ac PP. **SYNCOUT TB0** and **TRIGIN TB0** allow the sync out signal to be sent from channel A to channel B via the TB0 trigger bus.

Since channel B is set for gated waveform mode with **TRIGMODE GATE**, when the Sync Out signal goes from LOW to HIGH, the channel B waveform is output. When the Sync Out signal goes from HIGH to LOW, the channel B waveform is not output. Thus, channel B output occurs only when the channel A waveform is positive with respect to its DC offset (0 V in this case).

```
10   !file SYNCOUT
20   !
30   CLEAR 709                      !Clear HP 3245A
40   OUTPUT 709;"RST"               !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"           !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"             !Use channel A
80   OUTPUT 709;"  FREQ 1E3"        !Output freq = 1 kHz
90   OUTPUT 709;"  SYNCOUT TB0"     !Sync Out dest is TB0
100  OUTPUT 709;"  APPLY SQV 1"     !Output 1 V ac PP square wave
110  OUTPUT 709;"USE 100"           !Use channel B
120  OUTPUT 709;"  FREQ 10E3"       !Output freq = 10 kHz
130  OUTPUT 709;   TRIGMODE GATE"   !Set gated mode
140  OUTPUT 709;"  TRIGIN TB0"      !Trigger source is TB0
150  OUTPUT 709;"  APPLY ACV 2.5"   !Output 2.5 V ac PP sine wave
160  END
```

TBn?

Description

Trigger Bus Query. Returns the logic level ("0" or "1") of the specified trigger bus (TB0 or TB1).

Syntax

TB*n*?

Parameters

n Specifies which trigger bus level is read. A "0" specifies trigger bus 0 and a "1" specifies trigger bus 1.

Remarks

Data Returned

TBn? returns the logic level of the specified trigger bus (the trigger buses are normally high). A "0" is returned for a logic low level (0 V), while a "1" is returned for a logic high level (+5 V) on the trigger bus.

Data Destination

If the **MEM** command is used, the value returned by **TBn?** is stored in the specified variable or array. When **TBn?** is executed from the front panel, the response is displayed on the front panel display. When **TBn?** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

HP-IB Data Format

The HP 3245A returns numeric results in either ASCII or binary format (see the **OFORMAT** command). In the default ASCII format, **TBn?** returns integer results in 6-digit signed notation. In the binary format, **TBn?** returns integer results in 16-bit, 2's complement notation.

Related Commands

DRIVETBn,

Example

Example: Read Trigger Bus Level

This program reads the logic state of trigger bus 1 and displays the result on the controller CRT.

```
10  OUTPUT 709;"TB1?"       !Read logic state of trigger bus 1
20  ENTER 709;A             !Enter result (0 or 1)
30  PRINT "TB1 Level =";A   !Display result
40  END
```

If TB1 level is high, a typical return is:

```
    TB1 Level = 1
```

TERM/TERM?

Description

Set/Read Output Terminal. TERM selects the output terminal (front panel or rear panel) for the **USE** channel. **TERM?** returns the output terminal for the **USE** channel.

Syntax

TERM [*mode*]

TERM?

Parameters

mode The *mode* parameters follow. Power-on/reset/default *mode* = FRONT.

mode	Definition
OPEN/OFF	Disconnect all outputs.
FRONT	Front panel Output terminal.
REAR	Rear panel Output terminal.

Remarks

TERM Selects Output Terminals for USE Channel

TERM selects the output terminal destination for the **USE** channel. For example, if channel A is the **USE** channel, **TERM FRONT** selects the channel A front panel Output terminal, while **TERM REAR** selects the channel A rear panel Output terminal. (A channel's Front and Rear terminals cannot be selected at the same time.)

Query Command (TERM?)

The **TERM?** command returns the output destination terminal for the **USE** channel (channel A or B).

Related Commands

APPLYs, USE

Examples

Example: Selecting Output Terminal

This program selects the rear panel Channel A Output connector as the channel A output destination and returns the output terminal destination (REAR).

```
10   OUTPUT 709;"USE 0"                !Use channel A
20   OUTPUT 709;"TERM REAR"            !Use rear panel Output terminal
30   OUTPUT 709;"TERM?"                !Query output terminal
40   ENTER 709;A$                      !Enter terminal location
50   PRINT "Output Terminal = ";A$     !Display terminal location
60   END
```

TEST (TST)

Description

Confidence Test. **TEST** (or **TST**) performs a confidence test on the HP 3245A or on specified channel(s). **TEST** does not change the hardware or software state, but does set the output to 0 V on the specified channel(s).

Syntax

TEST [*ch*, [*ch*]]

Parameters

ch Channel parameter. Use **0** or **CHANA** for channel A or use **100** or **CHANB** for channel B. Executing **TEST** without the *ch* parameter tests the entire HP 3245A, including channels A and B and the front panel. Executing **TEST** *ch* tests only the specified channel.

Remarks

Data Returned

TEST returns "PASS" if all tests pass or "FAIL" if one or more tests fail. Any failures which occur during the test are displayed on the front panel and the first four errors are stored in the error register. Use the **ERR?** or **ERRSTR?** command to read the error register. Status register bit 5 (ERROR) is also set if any failures occur.

Channel Test

Executing **TEST** performs various tests on the specified channel(s), but does not change the hardware or software state of the channel. For the channel test, each internal register on the channel is addressed. If the registers do not respond, the test fails.

Front Panel Test

The front panel test is performed when the **TEST** command is executed without the *ch* parameter. After the front panel test is completed, both channels are tested.

Data Destination

When **TEST** is executed from the front panel, the response is displayed on the front panel display. When **TEST** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

Related Commands

DTEST, FTEST

Example

Example: Testing the HP 3245A (TEST)

This program tests the HP 3245A, channels A and B, and the front panel. If all tests pass, "Test Passed" appears on the controller CRT. If one or more tests fail, "Test Failed" appears on the controller CRT and the first four errors are displayed.

```
10   !file TEST
20   !
30   DIM Err$[60]               !Dimension controller array
40   OUTPUT 709;"TEST"          !Test HP 3245A and front panel
50   ENTER 709;A$               !Enter test result (pass/fail)
60   IF A$="FAIL" THEN          !Enter loop
70     PRINT "Test Failed"      !Display message if test fails
80     FOR I = 1 TO 4           !Error loop
90       OUTPUT 709;"ERRSTR?"   !Read error string
100      ENTER 709;Err$         !Enter string
110      PRINT Err$             !Display  string
120    NEXT I                   !Loop until error register is empty
130  ELSE
140    PRINT "Test Passed"      !Display message if test passes
150  END IF                     !End loop
160  END
```

TIME

Description

Read Time. Returns the current HP 3245A clock reading in number of seconds since midnight (see the **SET TIME** command).

Syntax

TIME

Parameters

None.

Remarks

Data Returned

The **TIME** command returns the current time in number of seconds since midnight in the range 0 through 86399.99 seconds.

Data Destination

If the **MEM** command is used, the value returned by **TIME** is stored in the specified variable or array. When **TIME** is executed from the front panel, the response is displayed on the front panel display. When **TIME** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

HP-IB Data Format

The HP 3245A returns numeric results in either ASCII or binary format (see the **OFORMAT** command). In the default ASCII format, **TIME** returns integer results in 6-digit signed notation. In the binary format, **TIME** returns integer results in 16-bit, 2's complement notation.

Converting Time Formats

HP 9000 Series 200/300 controllers use two commands to convert time formats: **TIME** and **TIME$**. The **TIME** command converts a formatted time-of-day string (hh:mm:ss) into a numeric value of seconds since midnight. The **TIME$** command converts the number of seconds value into a string representing the time-of-day format.

Related Commands

SET TIME

Example

Example: Setting/Reading Time Since Midnight (TIME)

This program sets the HP 3245A internal clock to 06:56:40 A.M. (25000 seconds past midnight) and verifies that the clock is running by reading the time five seconds after the clock is set (06:56:45).

```
10    !file TIME
20    !
30    CLEAR 709                 !Clear HP 3245A
40    OUTPUT 709;"RST"          !Reset HP 3245A
50    OUTPUT 709;"SCRATCH"      !Clear HP 3245A memory
60    !
70    OUTPUT 709;"SET TIME";(6*3600.)+(56*60)+40  !06:56:40 A.M.
80    WAIT 5                    !Wait five seconds
90    OUTPUT 709;"TIME"         !Read clock time
100   ENTER 709;A               !Enter time
110   PRINT TIME$(A)            !Convert time and display
120   END
```

TRG

Description

Pulse Trigger Bus. Pulses the trigger bus (TB0 or TB1) specified in the **DRIVETBn TRG** command when a **TRG** command is received by the HP 3245A.

Syntax

TRG

Parameters

None.

Remarks

TRG Performs the Same Action as the HP-IB TRIGGER Command

TRG performs the same action in the HP 3245A as the HP-IB TRIGGER (Group Execute Trigger) command. However, the TRIGGER command also triggers other instruments on the HP-IB bus, while **TRG** triggers only the HP 3245A.

Related Commands

DRIVETBn

Example

Example: Pulsing TB0 Trigger Bus (TRG)

This program sets channel A to output a 5 V ac PP ramp waveform when the **TRG** command is received. Since **TRIGMODE ARMWF** is set, the waveform is not output until a trigger is received from the source set by **TRIGIN** (trigger bus TB0 in this case). Since **DRIVETB0 TRG** is set, the trigger is generated when **TRG** is executed and pulses TB0. To generate the output waveform, run the program and press the Continue key when the program pauses.

```
10   !file TRG
20   !
30   CLEAR 709                     !Clear HP 3245A
40   OUTPUT 709;"RST"              !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"          !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"            !Use channel A
80   OUTPUT 709;"  TRIGMODE ARMWF" !Set "armed" mode
90   OUTPUT 709;"  DRIVETB0 TRG"   !Pulse TB0 when TRG received
100  OUTPUT 709;"  TRIGIN TB0"     !Trigger source is TB0
110  OUTPUT 709;"  FREQ 100"       !Frequency = 100 Hz
120  OUTPUT 709;"  APPLY RPV 5"    !Output 5 V ac PP ramp wave
130  PAUSE                         !Pause program
140  OUTPUT 709;"  TRG"            !Pulse TB0 trigger bus
150  END
```

Description

Triggered Frequency Mode. Allows fast change of AC waveform frequency (less than 140 μsec from a subroutine). Frequency change occurs on the high-to-low transition of the event specified by **TRIGIN** for the use channel.

Syntax

TRIGFREQ *ch, freq_mHz*

Parameters

ch **USE** Channel. 0 sets channel A, 100 sets channel B.

freq_mHz Frequency in millihertz (fractional part will be truncated).

Remarks

Using TRIGFREQ Command

Using **TRIGFREQ** allows frequency changes to be made faster than with **FREQ**, but eliminates features such a query, display monitoring (**MON STATE**), autoranging, parameter value checking and rounding, and automatic use channel. The event set by **TRIGIN** must be inactive (high) before **TRIGFREQ** is executed.

To enter the triggered-frequency mode, the desired waveform function must be selected with **APPLY ACV/ACI**, **APPLY RPV/RPI**, **APPLY SQV/SQI**, or **APPLY WFV/WFI**. Next, **TRIGFREQ** must be executed with an initial frequency value. Then, a second **APPLY** command (for the same function) or a **FASTAMP** command must be specified. (If the second **APPLY** command is **APPLY WFV/WFI**, the array must NOT be respecified.) When you are finished using the **TRIGFREQ** command, **RESET** is recommended to correct queries (**FREQ?**), display monitoring (**MON STATE**), autoranging, etc.

Related Commands

FASTAMP, FASTFREQ, FREQ

Example

Example: Triggered Frequency Changes (TRIGFREQ)

This program sweeps amplitude, DC offset, and frequency for both channel A and channel B. Note that using **TRIGFREQ** maintains precise phase relationship between the two channel outputs since the frequencies on both channels are changed simultaneously when **TRIGIN** is executed. If **FREQ** or **FASTFREQ** were used, some phase shift would occur sine the frequencies would not be simultaneously updated. This program requires about 800 μsec per iteration.

```
10    !file TRIGFREQ
20    !
30    ASSIGN @Source TO 709
40    N_points=3000                            !Number sweep points
50    OUTPUT @Source;"RST"                     !Reset HP 3245A
60    OUTPUT @Source;"SCRATCH"                 !Clear HP 3245A memory
70    !
80    !Define arrays, variables
90    !
100   OUTPUT @Source;"DIM F0(";N_points;")"        !Freq array
110   OUTPUT @Source;"INTEGER AMP0(";N_points;")"  !Ch A amplitude array
```

```
120  OUTPUT @Source;"INTEGER AMP1(";N_points;")"   !Ch B amplitude array
130  OUTPUT @Source;"INTEGER OFF0(";N_points;")"   !Ch A offset array
140  OUTPUT @Source;"INTEGER OFF1(";N_points;")"   !Ch B offset array
150  OUTPUT @Source;"INTEGER I,A"                  !Def INTEGER vars
160  !
170  !Define ch A and B amplitude and DC offset arrays
180  !
190  OUTPUT @Source;"SUB DEF_ARRAYS"               !Define arrays
200  OUTPUT @Source;"  FOR I=1 TO ";N_points
210  OUTPUT @Source;"    F0(I)=I*100."
220  OUTPUT @Source;"    AMP0(I)=888+888.*I/";N_points  !Ch A ampl
230  OUTPUT @Source;"    AMP1(I)=888"              !Ch B ampl
240  OUTPUT @Source;"    OFF0(I)=0"                !Ch A DC offset
250  OUTPUT @Source;"    OFF1(I)=660.*I/";N_points !Ch B DC offset
260  OUTPUT @Source;"  NEXT I"
270  OUTPUT @Source;"SUBEND"                       !End subroutine
280  OUTPUT @Source;"CALL DEF_ARRAYS"              !Call subroutine
290  !
300  !Set channel A triggering/output
310  !
320  OUTPUT @Source;"USE 0"                        !Use channel A
330  OUTPUT @Source;"TRIGIN HIGH"                  !Set TRIGIN HIGH
340  OUTPUT @Source;"TRIGOUT EXT"                  !Enable Trigger conn
350  OUTPUT @Source;"FREQ 0"                       !Hold at zero phase
360  OUTPUT @Source;"TRIGMODE ARMWF"               !Set armed mode
370  OUTPUT @Source;"APPLY ACV 1"                  !Apply sine wave
380  !
390  !Set channel B triggering/output
400  !
410  OUTPUT @Source;"USE 100"                      !Use ch A
420  OUTPUT @Source;"TRIGIN EXT"                   !External trigger
430  OUTPUT @Source;"FREQ 0"                       !Hold at zero phase
440  OUTPUT @Source;"TRIGMODE ARMWF"               !Set armed mode
450  OUTPUT @Source;"APPLY RPV 1"                  !Apply ramp wave
460  !
470  !Set initial values on channels A and B
480  !
490  OUTPUT @Source;"TRIGFREQ   0,F0(1)"           !ch A init freq
500  OUTPUT @Source;"TRIGFREQ 100,F0(1)"           !ch B init freq
510  OUTPUT @Source;"FASTAMP    0,AMP0(1),0"       !ch A ampl/offset
520  OUTPUT @Source;"FASTAMP  100,AMP1(1),0"       !ch B ampl/offset
530  !
540  !Change ch A and B amplitude, DC offset, frequency
550  !
560  OUTPUT @Source;"USE 0"                        !Use channel A
570  OUTPUT @Source;"SUB SWEEPA"                   !Define freq array
580  OUTPUT @Source;"  FOR I=1 TO ";N_points       !Sweep thru points
590  OUTPUT @Source;"    FASTAMP    0,AMP0(I),0"   !Update ch A ampl/off
600  OUTPUT @Source;"    FASTAMP  100,AMP1(I),0"   !Update ch B ampl/off
610  OUTPUT @Source;"    TRIGFREQ   0,F0(I)"       !ch A trig freq mode
620  OUTPUT @Source;"    TRIGFREQ 100,F0(I)"       !ch B trig freq mode
630  OUTPUT @Source;"    TRIGIN SGL"               !Change frequencies
640  OUTPUT @Source;"  NEXT I"
650  OUTPUT @Source;"SUBEND"
```

```
660   !
670   !Measure/display iteration time
680   !
690   T1=TIMEDATE
700   OUTPUT @Source;"SWEEPA"
710   T2=TIMEDATE
720   DISP (T2-T1)/N_points                                  !Display iteration time
730   STOP
740   END
```

A typical return (in µsec) is:

```
800
```

TRIGGER (GET)

Description

If **DRIVETBn TRG** is set, **TRIGGER (Group Execute Trigger)** triggers the specified trigger bus once, then holds triggering.

Syntax

TRIGGER 7
TRIGGER 709

Remarks

- **TRIGGER** performs the same action as the HP 3245A **TRG** command. However, the TRIGGER command also triggers other instruments on the HP-IB bus, while **TRG** triggers only the HP 3245A.

- If **INBUF ON** is set, **TRIGGER** executes immediately even if other commands are in the input buffer waiting for processing.

- If subroutine execution is suspended by an HP 3245A **PAUSE** command, **TRIGGER** resumes subroutine operation, but does not generate a trigger.

Example

```
TRIGGER 7        !Send Group Execute Trigger (GET)

TRIGGER 709      !Send Group Execute Trigger (GET) to
                 !device at address 09
```

TRIGIN/TRIGIN?

Description

Set/Read Trigger Input Source. **TRIGIN** selects the trigger source used to generate triggers on the **USE** channel. **TRIGIN?** returns the **TRIGIN** source for the **USE** channel.

Syntax

TRIGIN [*source*]

TRIGIN?

Parameters

source The *source* parameters follow. Power-on/reset *source* = HIGH, default *source* = SGL.

source	Definition
TB0	Trigger Bus 0. Trigger using the signal on the TB0 trigger bus.
TB1	Trigger Bus 1. Trigger using the signal on the TB1 trigger bus.
EXT	Trigger Connector. Trigger using the input to the front panel Trigger connector.
EXTBAR	Inverse of EXT. Trigger using the inverse of the input to the front panel Trigger connector.
LOW	Low Signal - Software Control. Used with the HIGH parameter to internally trigger.
HIGH	High Signal - Software Control. Used with the LOW parameter to internally trigger.
HOLD	Same as HIGH parameter.
SGL	Single Trigger. Assure a HIGH then change to LOW and back to HIGH to internally trigger.

Remarks

Selecting Trigger Source (TRIGIN)

TRIGIN *source* selects the trigger source used to generate triggers on the **USE** channel. Front panel triggering is selected with **TRIGIN EXT** or **EXTBAR**; software triggering with **TRIGIN HIGH**, **HOLD**, **LOW**, or **SGL**; and trigger bus triggering with **TRIGIN TB0** or **TB1**.

Front Panel Triggering (TRIGIN EXT/EXTBAR)

External triggers can be input to the Channel A and Channel B Trigger (I/O) ports on the front panel. When **TRIGIN EXT** or **EXTBAR** is set, the **USE** channel is triggered with an external (user-supplied) trigger into the Channel A or

Channel B Trigger (I/O) port on the front panel (the trigger input must be TTL-compatible). With **TRIGIN EXT**, the **USE** channel is triggered by the Trigger (I/O) port. With **TRIGIN EXTBAR**, the **USE** channel is triggered by the inverse of the Trigger (I/O) port.

Software Triggering (TRIGIN HIGH/HOLD/LOW/SGL)

You can internally (software) trigger the **USE** channel with **TRIGIN HIGH, HOLD, LOW,** or **SGL**. For example, **TRIGIN HIGH** or **TRIGIN HOLD** sets the **USE** channel input HIGH (+5V). Triggering in this manner is equivalent to setting **TRIGIN EXT** and inputting a +5V level to the **USE** channel Trigger (I/O) port. **TRIGIN LOW** sets the **USE** channel input LOW (0V). **TRIGIN SGL** first assures that the use channel is set HIGH, then sets the channel LOW and back to HIGH to single-trigger the channel.

Trigger Bus Triggering (TRIGIN TBn/DRIVETBn)

TRIGIN TB0 or **TRIGIN TB1** sets trigger bus triggering on the specified **USE** channel. **TRIGIN TB0** sets trigger bus TB0, while **TRIGIN TB1** sets trigger bus TB1 as the bus to be used. The **DRIVETBn** command selects the source to drive the specified trigger bus (TB0 or TB1). Use **DRIVETBn?** to read the source for the specified trigger bus.

With trigger bus triggering (**TRIGIN TB0** or **TRIGIN TB1** set), you can input an external trigger into the TB0 (I/O) or TB1 (I/O) port on the HP 3245A rear panel or you can use internal (software) triggering to trigger the specified **USE** channel. To do this, **DRIVETBn OFF** must be set. This disables the software drivers as well as any channel's drive (with **REFOUT** or **SYNCOUT**) to that trigger bus so that the TB0 or TB1 port can act as an input terminal. Then, you can input a TTL-compatible signal into the TB0 or TB1 port to trigger the specified **USE** channel (channel A or B).

Alternatively, you can use **DRIVETBn LOW, HIGH, SGL,** or **TRG** to internally drive the specified trigger bus. **DRIVETBn LOW** drives the specified trigger bus LOW (0V), while **DRIVETBn HIGH** drives the specified trigger bus HIGH (+5V). **DRIVETBn SGL** pulses the specified trigger bus (LOW then HIGH).

With **DRIVETBn TRG** set, the specified trigger bus is pulsed (LOW then HIGH) when the HP 3245A receives a **TRG** command or an HP-IB Group Execute Trigger (**GET**) command. Refer to **SYNCOUT** or **REFOUT** commands for more information on driving a trigger bus from a channel.

Query Command (TRIGIN?)

The **TRIGIN?** command returns the current **TRIGIN** *source* parameter setting (TB0, TB1, EXT, EXTBAR, LOW, HIGH, HOLD, or SGL) for the **USE** channel (channel A or B).

Related Commands

DRIVETBn, TRIGOUT, USE

Example

Example: Using External Triggering (TRIGIN)

This program uses **TRIGIN EXT** to trigger the channel B output with the Sync Out pulse from channel A. This causes the waveform on channel B to be 180° out of phase with the channel A waveform. Channels A and B are configured to output a 1 V ac PP, 500 Hz sine waveform.

Since **TRIGMODE ARMWF** and **TRIGIN EXT** are set on channel B, channel B will not generate an output until an external trigger is received on the channel. When the channel A output is positive, the Sync Out pulse is HIGH (+5V) and when the channel A output is negative, the Sync Out pulse is LOW.

Thus, channel B is triggered when the Sync Out pulse goes from HIGH to LOW (at the 180° point of the channel A waveform). As a result, the channel B waveform is 180° out of phase with the channel A waveform. To ensure that the 180° phase relationship is maintained and that both outputs remain at the same frequency, the channels use the channel A frequency reference.

```
10   !file TRIGIN
20   !External connections - connect a BNC cable between SYNC OUT of
30   !channel A and TRIGGER (I/O) of channel B.
40   !
50   CLEAR 709                        !Clear HP 3245A
60   OUTPUT 709;"RST"                 !Reset HP 3245A
70   OUTPUT 709;"SCRATCH"             !Clear HP 3245A memory
80   !
90   OUTPUT 709;"USE 0"               !Use channel A
100  OUTPUT 709;"  FREQ 500"          !Frequency = 500 Hz
110  OUTPUT 709;"  REFOUT EXT"        !Ref freq dest is Freq Ref conn
120  OUTPUT 709;"  APPLY ACV 1"       !Output 1 V ac PP sine wave
130  OUTPUT 709;"USE 100"             !Use channel B
140  OUTPUT 709;"  TRIGMODE ARMWF"    !Set sync mode on ch B
150  OUTPUT 709;"  TRIGIN EXT"        !Trigger conn is trigger source
160  OUTPUT 709;   REFIN EXT"         !Freq Ref conn is input source
170  OUTPUT 709;"  FREQ 500"          !Output freq is 500 Hz
180  OUTPUT 709;"  APPLY ACV 1"       !Output 1 V ac PP sine wave
190  END
```

TRIGMODE/TRIGMODE?

Description

Set/Read Trigger Mode. **TRIGMODE** selects one of three AC triggering modes for the **USE** channel. **TRIGMODE?** returns the current triggering mode on the **USE** channel. **TRIGMODE** has no effect on DC outputs.

Syntax

TRIGMODE *mode*

TRIGMODE?

Parameters

mode The *mode* parameters follow. Power-on/reset *mode* = OFF.

mode	Definition
OFF	Disables all triggering modes.
ARMWF	Selects the synchronized mode.
GATE	Selects the gated waveform mode.
DUALFR	Selects the dual frequency mode.

Remarks

Selecting Trigger Mode (TRIGMODE)

TRIGMODE *mode* selects one of three AC triggering modes for the **USE** channel. **TRIGMODE OFF** disables all triggering modes; **TRIGMODE ARMWF** selects synchronized (armed) mode; **TRIGMODE GATE** selects gated output mode; and **TRIGMODE DUALFR** selects dual-frequency mode. The power-on *mode* is **TRIGMODE OFF.**

Synchronized Mode (TRIGMODE ARMWF)

TRIGMODE ARMWF allows AC waveforms from both channels to be synchronized. In this mode, the waveform begins outputting when a high-to-low edge is received. This allows the synchronization of two waveforms, since when a trigger from the source set by **TRIGIN** is simultaneously received on both channels, AC waveforms begin outputting synchronously from both channels. Synchronized mode is convenient for generating multiphase AC waveforms and precise harmonics.

In synchronized mode, one channel is designated as a "master" and one or more other channels are designated as "slaves". Each channel (master or slave) generates a reference frequency (1073741.824 Hz) based on its internal reference oscillator.

In synchronized mode, each slave's oscillator locks on to the reference frequency generated by the master. When the master and the slaves are triggered (all channels must be triggered simultaneously), each channel begins outputting its waveform at the selected frequency and phase angle. At power-on/reset, channel B is slaved to channel A.

When synchronized mode has been enabled, use an **APPLY** command to set the waveform function (sine, ramp, square, or arbitrary) on each channel and its assigned frequency (**FREQ**) and phase angle (**PANG**). When the master and slaves are simultaneously triggered by a trigger from the **TRIGIN** source, each channel begins outputting its defined waveform. Note that all channels MUST be triggered at the same time.

In synchronized mode, the **APPLY** commands, **ARANGE ON**, **DCOFF**, **DUTY**, **FREQ**, **IMP**, **PANG**, **RANGE**, and **TRIGMODE ARMWF** reset the waveforms from the master and the slaves to their starting phase. After one of these commands is executed, the waveform returns to its assigned phase angle and waits for another trigger before resuming. (Sending an **ARANGE**, **IMP**, or **RANGE** command to the same range will not reset the waveforms to their starting phase.)

Gated Mode (TRIGMODE GATE)

AC waveforms can be gated by using the **TRIGMODE GATE** command. Gated mode is convenient for generating waveforms which must start and stop at precisely the zero phase point. With gated mode, when the signal on the trigger event selected by **TRIGIN** is LOW (0V), the channel generates the waveform specified by **APPLY**. When the trigger signal goes HIGH (+5 V), the waveform output continues until it reaches its zero phase point and then stops.

Zero phase is defined as the positive-going, zero-crossing point of a sine, ramp, or square waveform (relative to the DC offset). For arbitrary waveforms, zero degrees is defined as the first of the 2048 points required to define the waveform. At power-on, gated mode is disabled (**TRIGMODE OFF**) and the waveform is continuously generated (no trigger is required).

Dual-Frequency Mode (TRIGMODE DUALFR)

With dual-frequency mode, an AC waveform (sine, ramp, square, or arbitrary) can be modulated between two frequencies. An **APPLY** command sets the waveform to one of the two assigned frequencies (**FREQ** command) based on the logic level of the trigger event. You can use **TRIGMODE DUALFR** to enable modulation of AC waveforms between two frequencies. This mode is convenient to generate precise frequency changes or to hold a waveform at a specified frequency determined by the logic level of the selected trigger event.

The two waveform frequencies are set by the **FREQ** *freq_1* [*,freq_2*] command, where *freq_1* is the output frequency when the event selected by **TRIGIN** is HIGH (+5V) and *freq_2* is the output frequency when the trigger event is LOW (0V). For sine, square, and arbitrary waveforms, both frequencies must be in the range 0 to 1 MHz. For ramp waveforms, both frequencies must be in the range 0 to 100 kHz. At power-on, both frequencies are set to 1000 Hz.

NOTE

Executing **APPLY** *commands*, **ARANGE ON**, **DCOFF**, **DUTY**, **FREQ**, **IMP**, **RANGE**, *or* **TRIGMODE** *may cause the output waveform to momentarily change to the other of the two frequencies.*

Query Command (TRIGMODE?)

The **TRIGMODE?** command returns the current *mode* setting (OFF, ARMWF, GATE, DUALFR) for the **USE** channel.

Related Commands

APPLYs, FREQ, PANG, REFIN, REFOUT, TRIGIN, USE

Examples

Example: Generating Armed Waveform (TRGMDE_1)

This program shows how **TRIGMODE ARMWF** arms (but does not trigger) channel A. This prevents the waveform from being output until a trigger from **TRIGIN** *source* is received. When the program executes, channel A is armed and configured to generate a 100 kHz square wave @ 5 V ac PP and 2.5 VDC offset.

Since the channel was armed by **TRIGMODE ARMWF**, the waveform is not generated until **TRIGIN SGL** is executed, even though **APPLY SQV 5** is executed. Note that the program is paused (line 120) to allow changes to the circuit or device to receive the output, as required. To generate the waveform, run the program and then press the Continue key.

```
10   !file TRGMDE_1
20   !
30   CLEAR 709                    !Clear HP 3245A
40   OUTPUT 709;"RST"             !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"         !Clear HP 3245A memory
60   !
70   OUTPUT 709;"USE 0"           !Use channel A
80   OUTPUT 709;"  TRIGMODE ARMWF" !Set "armed" mode
90   OUTPUT 709;"  FREQ 100E3"    !Output freq is 100 kHz
100  OUTPUT 709;"  DCOFF 2.5"     !2.5 VDC offset
110  OUTPUT 709;"  APPLY SQV 5"   !Output square wave @ 5 V ac PP
120  PAUSE                        !Pause program
130  OUTPUT 709;"  TRIGIN SGL"    !Single trigger channel A
140  END
```

Example: Generating Gated Waveform (TRGMDE_2)

This program enables gated waveform mode on the channel A. When the external input to the channel is LOW (0V), the channel outputs a 60 Hz sine wave @ 1.5 V ac PP. When the external input is HIGH (+5V), the sine waveform continues to its 0 phase point and then stops.

```
10   !file TRGMDE_2
20   !External connections - connect a power supply to the 3245A
30   !TRIGGER (I/O) BNC (channel A).  Vary supply between 0V and +5V.
40   !
50   CLEAR 709                    !Clear HP 3245A
60   OUTPUT 709;"RST"             !Reset HP 3245A
70   OUTPUT 709;"SCRATCH"         !Clear HP 3245A memory
80   !
90   OUTPUT 709;"USE 0"           !Use channel A
100  OUTPUT 709;"  TRIGMODE GATE" !Set gated waveform mode
```

```
110  OUTPUT 709;"  TRIGIN EXT"      !Trigger conn is trig source
120  OUTPUT 709;"  FREQ 60"         !Frequency = 60 Hz
130  OUTPUT 709;"  APPLY ACV 1.5"   !Output sine wave  @ 1.5 V ac PP
140  END
```

Example: Generating Dual-Frequency Waveform (TRGMDE_3)

This program shows how the output frequency can be changed based on the level of the trigger signal. When the program executes, a subroutine is downloaded which outputs a 100 Hz or 1 kHz sine wave @ 1 V ac PP, based on the trigger signal level. Since the power-on value of **TRIGIN** is HIGH, the first frequency listed in the **FREQ** command (100 Hz) is output. Then, when **TRIGIN LOW** is executed, the second frequency in the **FREQ** command (1 kHz) is output.

NOTE

This program loops continuously from line 120 to line 160. To exit the loop and stop the program, press the front panel **Local** *key and reset the instrument from the HP 3245A front panel.*

```
10   !file TRGMDE_3
20   !
30   CLEAR 709                        !Clear HP 3245A
40   OUTPUT 709;"RST"                 !Reset HP 3245A
50   OUTPUT 709;"SCRATCH"             !Clear HP 3245A memory
60   !
70   OUTPUT 709;"SUB DUAL_FREQ"       !Begin subroutine
80   OUTPUT 709;"  USE 0"             !Use channel A
90   OUTPUT 709;"    TRIGMODE DUALFR" !Set dual frequency mode
100  OUTPUT 709;"    FREQ 100,1E3"    !Output 100 Hz or 1000 Hz
110  OUTPUT 709;"    APPLY ACV 1"     !Output 1 V ac sine wave
120  OUTPUT 709;"    WHILE 1"         !Begin continuous loop
130  OUTPUT 709;"      TRIGIN HIGH"   !Set freq to 100 Hz
140  OUTPUT 709;"      WAIT 1"        !Maintain 100 Hz for 1 sec
150  OUTPUT 709;"      TRIGIN LOW"    !Set freq to 1 kHz
160  OUTPUT 709;"      WAIT 1"        !Maintain 1 kHz for 1 sec
170  OUTPUT 709;"    END WHILE"       !End loop
180  OUTPUT 709;"SUBEND"              !End subroutine
190  OUTPUT 709;"RUN DUAL_FREQ"       !Execute subroutine
200  END
```

TRIGOUT/TRIGOUT?

Description

Enable/Read Trigger Output. **TRIGOUT** enables or disables trigger outputs from the **USE** channel Trigger connector when the channel is triggered with **TRIGIN HIGH, HOLD, LOW,** or **SGL**. **TRIGOUT?** returns the current setting for the **USE** channel Trigger connector.

Syntax

TRIGOUT *control*

TRIGOUT?

Parameters

control The *control* parameters follow. Power-on/reset *control* = OFF.

control	Description
EXT	Enables outputs from the Trigger connector.
OFF	Disables outputs from the Trigger connector.

Remarks

Hardware Input Triggers Disable Trigger Connector

Outputs from the Trigger connector are automatically disabled when **TRIGIN TB0, TB1, EXT,** or **EXTBAR** is selected. Similarly, if **TRIGIN TB0, TB1, EXT** or **EXTBAR** is selected and **TRIGOUT EXT** is then selected, **TRIGIN HIGH** is automatically set.

Query Command (TRIGOUT?)

The **TRIGOUT?** command returns the current setting of the **USE** channel Trigger connector.

Related Commands

TRIGIN, USE

Example

Example: Triggering Channels Simultaneously (REFIN_OUT)

This program uses **REFIN** and **REFOUT** to lock channel B (slave) output to channel A (master) output. Both channels output a 5 V ac PP ramp waveform. Since the Trigger ports of the two channels are externally connected, a trigger on channel A is also routed to channel B so that both channels are simultaneously triggered.

Since **REFOUT EXT** and **REFIN EXT** are set, the reference frequency on channel A is routed (internally) via the Freq Ref connector on the rear panel (no external connection is required). This prevents any drift in relative frequency and ensures that the two channel outputs are exactly at the same frequency. **TRIGIN LOW** triggers both channels simultaneously to begin synchronous outputs.

```
10   !file REFIN_OUT
20   !
3    !External connections - connect a BNC cable between the
4    !Trigger (I/O) ports of channels A and B.
5    !
60   CLEAR 709                      !Clear HP 3245A
70   OUTPUT 709;"RST"               !Reset HP 3245A
80   OUTPUT 709;"SCRATCH"           !Clear HP 3245A memory
90   !
100  OUTPUT 709;"USE 0"             !Use channel A
110  OUTPUT 709;"  TRIGMODE ARMWF"  !Set synch mode on ch A
120  OUTPUT 709;"  TRIGOUT EXT"     !Enable trigger out from ch A
130  OUTPUT 709;"  REFOUT EXT"      !Freq Ref conn is destination
140  OUTPUT 709;"  FREQ 5E3"        !Output freq is 5 kHz
150  OUTPUT 709;"  APPLY RPV 5"     !Output 5 V ac PP ramp wave
160  !
170  OUTPUT 709;"USE 100"           !Use channel B
180  OUTPUT 709;"  TRIGMODE ARMWF"  !Set synch mode on ch B
190  OUTPUT 709;"  TRIGIN EXT"      !Trigger conn is trigger source
200  OUTPUT 709;   REFIN EXT"       !Freq Ref conn is input source
210  OUTPUT 709;"  FREQ 5E3"        !Output freq is 5 kHz
220  OUTPUT 709;"  APPLY RPV 5"     !Output 5 V ac PP ramp wave
230  !
240  OUTPUT 709;"TRIGIN LOW"        !Trigger both chs simultaneously
250  END
```

USE/USE?

Description

Set/Read Use Channel. **USE** selects the channel (A or B) to receive subsequent commands. **USE?** returns the current **USE** channel.

Syntax

USE *ch*

USE?

Parameters

ch Selects the channel (A or B) to receive subsequent commands. **USE 0** or **USE CHANA** selects channel A. **USE 100** or **USE CHANB** selects channel B. Power-on/reset *ch* = channel A (**USE 0**).

Remarks

USE Within Subroutines

When a subroutine is executed, its **USE** *ch* is initially set to the same channel as previously defined in the main program. The subroutine may then define a different **USE** *ch*. If **USE** is executed within a **CALL** subroutine, the *ch* designation in the subroutine is retained when the subroutine finishes execution.

If **USE** is executed within a **RUN** subroutine, the HP 3245A uses that **USE** *ch* designation ONLY in the subroutine. After the subroutine finishes execution, the assignment reverts back to the value assigned before the subroutine was run.

Front Panel vs. HP-IB USE Channels

Commands originating from the front panel or HP-IB may or may not have the same **USE** channel. If a **USE** channel is assigned from the front panel, all subsequent commands entered from the front panel (or from HP-IB) will be directed to that channel. If a different **USE** channel is assigned over HP-IB all subsequent commands sent over HP-IB will be directed to that channel.

Query Command (USE?)

The **USE?** command returns the current **USE** channel (channel A or B). **USE?** returns "0" if channel A is the **USE** channel or returns "100" if channel B is the **USE** channel.

Related Commands

CALL, RUN

Examples

Example: Selecting Channel B as USE Channel.

At power-on, channel A is selected as the **USE** channel. This program line sets channel B as the **USE** channel. Note that **USE CHANB** could also be used.

```
10  OUTPUT 709;"USE 100"   !Use channel B
```

Description

Read Variable/Array. Returns the value of the specified variable, expression, or array.

Syntax

```
         variable
VREAD    expression
         array_name [element]
```

Parameters

variable	Variable read.
expression	Any valid number or numeric expression (enclosed in parentheses).
array_name [*element*]	Array name/element read.

Remarks

Reading Variable/Array Value Using VREAD

Use **VREAD** *variable* to read the value of a variable. For example, **VREAD A** reads the value of variable A. Use **VREAD** *array_name* [(*element*)] to read the value of an array or a specified element. When *element* is not specified, **VREAD** returns the values of all elements in the array.

When *element* is specified, **VREAD** returns the value of the specified element. For example, **DIM VOUT(3)** defines a 4-element array. **VREAD VOUT(2)** returns the value of element #2, while **VREAD** would return the value of each of the elements in VOUT.

Turn Memory Mode OFF Before Using VREAD

VREAD transfers variable or array values from memory to the output buffer. Always turn memory mode OFF (**MEM OFF**) before executing **VREAD**. Otherwise, **VREAD** attempts to write data into memory and an error occurs Memory mode is automatically turned off as required for variables, but not for arrays.

Data Destination

If **MEM** is used, the value returned by **VREAD** is stored in the specified variable or array. When **VREAD** is executed from the controller, the response is sent to the output buffer in the default ASCII format.

HP-IB Data Format

The HP 3245A returns numeric results in either ASCII or binary format (see the **OFORMAT** command). In the default ASCII format, **VREAD** returns REAL variables in 8-digit scientific notation and INTEGER variables in 6-digit format. (Array elements are separated by carriage return and line feed.) In binary format, **VREAD** returns REAL variables in IEEE-754 64-bit format and INTEGER variables in 16-bit format.

Related Commands

DISP, DSP, FETCH

Examples

Example: Reading Array Values Using MEM/VREAD

This program defines array A as a 3-element INTEGER array and stores the HP 3245A HP-IB address and duty cycle in the first two elements of the array (element 3 remains at 0). Then, **VREAD A** reads the data stored in array A. Since A is an array and since memory mode is NOT automatically turned off for arrays, **MEM OFF** (line 100) must be used to prevent an error when **VREAD** is executed.

```
10   DIM A(0:2)
20   CLEAR 709                   !Clear HP 3245A
30   OUTPUT 709;"RST"            !Reset HP 3245A
40   OUTPUT 709;"SCRATCH"        !Clear HP 3245A memory
50   !
60   OUTPUT 709;"INTEGER A(2)"   !Define A as INTEGER array
70   OUTPUT 709;"MEM A"          !Store data in array A
80   OUTPUT 709;"ADDRESS?"       !Query HP 3245A address
90   OUTPUT 709;"DUTY?"          !Query duty cycle
100  OUTPUT 709;"MEM OFF"        !Turn memory mode off
110  OUTPUT 709;"VREAD"          !Read array A values
120  ENTER 709;A(*)              !Enter values
130  PRINT A(*)                  !Display values
140  END
```

For a factory-configured instrument, a typical return is:

```
9    50    0
```

Example: Reading Array Values Using FILL/VREAD

This program uses **VREAD** to read array VOUT which contains four stored readings to be output. This example is similar to the previous program, except that **FILL** is used to fill the array.

```
10   DIM Rdgs(0:3)                     !Dimension controller array
20   OUTPUT 709;"DIM VOUT(3)"          !Dimension array VOUT
30   OUTPUT 709;"FILL VOUT -5,0,5,10"  !Enter array values
40   OUTPUT 709;"VREAD VOUT"           !Read array element values
50   ENTER 709; Rdgs(*)                !Enter results
60   PRINT Rdgs(*)                     !Display results
70   END
```

A typical return is:

```
-5    0    5    10
```

Description

Wait. The HP 3245A waits the specified number of seconds before continuing.

Syntax

WAIT *seconds*

Parameters

seconds

Specifies the time (in seconds) that the HP 3245A pauses. Valid range = 0 seconds through 21,474,836 seconds (about 248 days). Power-on/reset *seconds* = 0.

Remarks

WAIT Time Resolution

Resolution for the **WAIT** command is 10μs for values ≤30 msec and 10 msec for values >30 msec.

WAIT Within RUN Subroutine

If **WAIT** is specified within a RUN subroutine, the wait occurs within the subroutine only.

Related Commands

WAITFOR

Example

Example: Using the WAIT Command

This program statement pauses the HP 3245A for 2 msec.

```
100  OUTPUT 709;"WAIT 2E-3"  !Pause for 2 msec.
```

WAITFOR

Description

Wait for Event. Causes the HP 3245A to wait until a trigger edge is received from the specified source.

Syntax

WAITFOR *source, edge*

Parameters

source

The *source* parameters follow.

source	Definition
TB0	Wait for edge on backplane trigger bus 0.
TB1	Wait for edge on backplane trigger bus 1.

edge

The *edge* parameters follow.

edge	Definition
HL	Wait for high-to-low edge.
LH	Wait for low-to-high edge.

Remarks

Related Commands

WAIT

Example

Example: Wait For Edge

This program statement causes the HP 3245A to wait for a falling (HL) edge on trigger bus TB0.

```
10    OUTPUT 709;"WAITFOR TB0,HL"    !Wait for HL edge on TB0
```

Example: Wait For Pulse

This program line causes the HP 3245A to wait for a high-to-low-to-high pulse on trigger bus TB1.

```
10    OUTPUT 709;"WAITFOR TB1,HL;WAITFOR TB1,LH"
```

Description

While-End While Loop. Defines a loop within a subroutine which is repeated as long as the numeric expression is TRUE.

Syntax

WHILE *expression*

program segment

END WHILE

Parameters

expression A numeric expression that is evaluated as TRUE if nonzero, or FALSE if evaluated as zero.

Remarks

WHILE..END WHILE Valid Only in Subroutines

WHILE may be used only within HP 3245A subroutines and the subroutine must be called (**CALL** command) or run (**RUN** command) to execute the commands.

Using the WHILE..END WHILE Command

The **WHILE..END WHILE** operation depends on the result of a test performed at the start of the loop. If the test is TRUE (not equal to zero), the program segment between the **WHILE** and **END WHILE** statements is executed and a branch is made back to the **WHILE** statement. If the test is FALSE (equal to zero), program execution continues with the first statement after the **END WHILE** statement.

Related Commands

FOR. . .NEXT, IF. . .END IF, SUB, SUBEND

Example

Example: Using a WHILE...END WHILE Loop (WHILE)

This program uses subroutine PWRTWO with a **WHILE** loop to display the powers of 2 less than 1000 (1, 2, . . ., 512) on the front panel display.

```
10   !file WHILE
20   !
30   CLEAR 709
40   OUTPUT 709;"RST"
50   OUTPUT 709;"SCRATCH"
60   !
70   OUTPUT 709;"SUB PWRTWO"          !Define subroutine PWRTWO
80   OUTPUT 709;"  I=1"               !I = 1
90   OUTPUT 709;"  WHILE I <= 1000"   !Continue until 2*I >= 1000
100  OUTPUT 709;"    DISP I"          !Display value of I
110  OUTPUT 709;"    I=(2*I)"         !Assign next value to I
120  OUTPUT 709;"    WAIT 1"          !Wait 1 second
130  OUTPUT 709;"  END WHILE"         !End when 2*I >= 1000
140  OUTPUT 709;"  DISP 'Done'"       !Display message when done
```

```
150  OUTPUT 709;"SUBEND"          !End subroutine
160  OUTPUT 709;"CALL PWRTWO"     !Call subroutine PWRTWO
170  END
```