REMOTE CONTROL AND PROGRAMMING REFERENCE

for the FLUKE 96

ScopeMeter Test Tool

==========

This file contains remote control and programming information for the FLUKE 96 with use of the PM9080/001 Isolated Optical to RS232 Adapter/Cable.

It consists of the following chapters:

1. INSTALLING THE PM9080/001

2. INTRODUCTION TO PROGRAMMING

3. COMMAND REFERENCE

APPENDIXES

APPENDIX A	ACKNOWLEDGE DATA
APPENDIX B	STATUS DATA

1. INSTALLATION OF THE PM9080/001

- Connect the PM9080/001 to the RS232 port of the computer as indicated in the PM9080/001 Instruction Manual. If necessary, use the 9-pin to 25-pin adapter and the 25-pin gender changer included in the PM9080/001 shipment.

- Hook the PM9080/001 cable to the ScopeMeter test tool as indicated in the PM9080/001 Instruction Manual.

- Turn on the computer and the ScopeMeter test tool.

- Make sure that the communication settings match for the RS232 port of the computer and the ScopeMeter test tool.

  After power-on, the default settings of the ScopeMeter test tool are as follows:

  1200 baud, No parity, 8 data bits, 1 stop bit

  You can modify these settings with the PC (Program Communication) command. See chapter 3 COMMAND REFERENCE.

  You can modify the computer RS232 port settings to match the above ScopeMeter test tool settings with the following DOS command:

  MODE COM1:1200,N,8,1

  This command assumes that COM1 is the RS232 port used on the computer. Replace COM1 in the above command with COM2, COM3, or COM4 if one of these ports is used. You can place this command in the computer startup file AUTOEXEC.BAT so that the default settings for the computer are the same as for the ScopeMeter test tool. If you want to use a higher data transfer speed (baud rate), let your QBASIC program change the settings for both the computer and the ScopeMeter test tool. See the example under the PC (Program Communication) command in chapter 3 COMMAND REFERENCE.

2. INTRODUCTION TO PROGRAMMING

** Basic Programming Information **

When you have installed the PM9080/001 as described in the previous chapter, you can control the ScopeMeter test tool from the computer with simple communication facilities, such as GWBASIC, QuickBASIC and QBASIC (programming languages from Microsoft Corporation).

All examples given in this manual are in the QBASIC language but will also run in QuickBASIC. QuickBASIC allows you to make executable files from programs so you can start such programs directly from DOS.
It is assumed that you have knowledge of these programming languages. QBASIC is supplied with Microsoft Operating System MS-DOS 5.0 and higher, and has an 'on-line' Help function.

Features of the syntax and protocol for the ScopeMeter test tool are as follows:

- Easy input format with a 'forgiving' syntax:
  All commands consist of two characters that can be UPPER or lower case.
  Parameters that sometimes follow the command may be separated from it by one or more separation characters.

- Strict and consistent output format:
  Alpha character responses are always in UPPERCASE.
  Parameters are always separated by a comma ("," = ASCII 44, see Appendix D).
  Responses always end with the carriage return code (ASCII 13). Because the carriage return code is a non-visible character (not visible on the screen or on paper), this character is represented as <cr> in the command syntax.
- Synchronization between input and output:
  After receipt of every command, the ScopeMeter test tool returns an acknowledge character (digit) followed by the carrige return code (ASCII 13). This indicates that the command has been successfully received and executed.
  The computer program must always read this acknowledge response before sending the next command to the ScopeMeter test tool.

** Commands sent to the ScopeMeter test tool **

All commands for the ScopeMeter test tool consist of a header made up of two alpha characters sometimes followed by parameters. Example:

RI	This is the Reset Instrument command. It resets the ScopeMeter test tool.

Some of the commands are followed by one or more parameters to give the ScopeMeter test tool more information. Example:

PC2400,N,8,1	This is the Program Communication command. This command requires four parameters. The parameters are separated by a comma, which is called the Program Data Separator. You may also use spaces (SP = ASCII 32) or tabs (HT = ASCII 9) to separate parameters. For example, the previous example may also be written as follows: PC 2400 N 8 1

You may use several spaces or tabs to separate parameters but if a comma is used for separation, you may use only one comma between the parameters. Also refer to the section 'Data Separators'.

A code at the end of each command tells the ScopeMeter test tool that the command is ended. This is the carriage return code (ASCII 13) and is called the Program Message Terminator. This code is needed to indicate to the ScopeMeter test tool that the command is completed so it can start executing the command. Also refer to the section 'Command and Response Terminators'.

** Responses received from the ScopeMeter test tool **

After each command sent to the ScopeMeter test tool, there is an automatic response from it, indicated as <acknowledge> (which you MUST input), to let the computer know whether or not the received command has been successfully executed. Refer to the 'Acknowledge' section below.

There are several commands that ask the ScopeMeter test tool for response data. Such commands are called Queries. Example:

ID — This is the IDentification query, which asks for the model number and the software version of the ScopeMeter test tool.

When the ScopeMeter test tool has received a query, it sends the <acknowledge> reply as it does after any command, but now it is followed by the queried response data.

The format of the response data depends upon which query is sent. When a response consists of different response data portions, these are separated with commas (ASCII code 44). Also refer to the section 'Data Separators'.

All response data, <acknowledge> as well as following (queried) response data are terminated with the carriage return code (<cr> = ASCII 13). Also refer to the section 'Command and Response Terminators'.

** Acknowledge **

After receiving of a command, the ScopeMeter test tool automatically returns the <acknowledge> response to let the computer know whether or not the received command has been successfully executed.
This response is a one-digit number followed by <cr> as response terminator. If <acknowledge> is 0, it indicates that the ScopeMeter test tool has successfully executed the command. If the command was a query, the <acknowledge><cr> response is immediately followed by the queried response data terminated with <cr>.
If <acknowledge> is 1 or higher, it indicates that the ScopeMeter test tool has not executed the command successfully. In that case, if the command was a query, the <acknowledge><cr> response is NOT followed by any further response data.
There can be several reasons for a non-zero <acknowledge> response. For more information see Appendix A.
In case of an error you can obtain more detailed status information by using the ST (STATUS) query.

Note: YOU MUST ALWAYS INPUT <acknowledge>, EVEN WHEN THE COMMAND WAS NOT A QUERY.

** Data Separators **

Data Separators are used between parameters sent to the ScopeMeter test tool and between values and strings received from the ScopeMeter test tool. The following list shows the data separators that can be used:

- Program Data Separator

Name	Character	ASCII Value Decimal	Comments
comma	,	44	Single comma allowed
space	SP	32	Multiple spaces allowed
tab	HT	9	Multiple tabs allowed

- Response Data Separator

Name	Character	ASCII Value Decimal	Comments
comma	,	44	

** Command and Response Terminators **
(Message Terminators)

- Command (Program Message) Terminators

  A code is needed at the end of each command to tell the ScopeMeter test tool that the command is ended, and that it can start executing the command. This code is called the Program Message Terminator. The code needed for the test tool is carriage return (ASCII code 13 decimal).
  Notes:

  1. The carriage return code is a non-visible ASCII character. Therefore this code is represented as <cr> in the Command Syntax and Response Syntax lines given for each command.
  2. The QBASIC programming language, which is used for all program examples, automatically adds a carriage return to the end of the command output. (In the QBASIC language, this is the PRINT #.... statement.)

  After <cr> is recognized by the ScopeMeter test tool, the entered command is executed. After EACH command the ScopeMeter test tool returns <acknowledge><cr> to the computer to signal the end of the command processing (also see the section 'Acknowledge'.)

- Response (Message) Terminators

  The response from the ScopeMeter test tool ends with a carriage return (ASCII 13). This is indicated as <cr> in the Response Syntax for each command.

** Typical program sequence **
An example

A typical program sequence consists of the following user actions:

1. Set the communication parameters for the RS232 port of the computer to match the ScopeMeter test tool settings.

2. Output a command or query to the test tool.

3. Input the acknowledge response from the test tool.

   If the response value is zero, go to step 4.

   If the response value is non-zero, the test tool did not execute the previous command. Read the error message from the following acknowledge subroutine, recover the error, and repeat the command or query. (This is not shown in the following program example.)

4. If a query was output to the test tool, input its response.

5. The sequence of points 2, 3, and 4 may be repeated for different commands or queries.

6. Close the communication channel.

Refer to the program example on the next page.

```
'Example of a typical program sequence:

'*****************  Begin example program  ***************

OPEN "COM1:1200,N,8,1,CS,DS,RB2048" FOR RANDOM AS #1

     'This QBASIC program line sets the parameters for the
     'RS232 port (COM1 on the Computer) to match the test
     'tool power-on default settings. It also opens a
     'communication channel (assigned #1) for input or output
     'through the COM1 port. Your test tool must be connected
     'to this port. "RB2048" sets the size of the computer
     'receive buffer to 2048 bytes to prevent buffer overflow
     'during communication with the ScopeMeter test tool.

PRINT #1, "ID"

     'Outputs the IDENTITY command (query) to the test tool.

GOSUB Acknowledge

     'This subroutine inputs the acknowledge response from
     'the test tool and displays an error message if the
     'acknowledge value is non-zero.

INPUT #1, Response$

     'This inputs the response data from the IDENTITY query.

PRINT Response$

     'Displays the queried data.

CLOSE #1

     'This closes the communication channel.

END

     'This ends the program.

'
```

```
'*****************  Acknowledge subroutine  ******************
'Use this subroutine after each command or query sent to the
'ScopeMeter test tool. This routine inputs the acknowledge
'response from the test tool. If the response is non-zero,
'the previous command was not correct or was not correctly
'received by the test tool. Then an error message is
'displayed and the program is aborted.

Acknowledge:
INPUT #1, ACK           'Reads acknowledge from test tool.
IF ACK <> 0 THEN
   PRINT "Error "; ACK; ": ";
   SELECT CASE ACK
     CASE 1
        PRINT "Syntax Error"
     CASE 2
        PRINT "Execution Error"
     CASE 3
        PRINT "Synchronization Error"
     CASE 4
        PRINT "Communication Error"
     CASE IS < 1
        PRINT "Unknown Acknowledge"
     CASE IS > 4
        PRINT "Unknown Acknowledge"
   END SELECT
   PRINT "Program aborted."
   END
END IF
RETURN

'******************  End example program  *****************
```

3. COMMAND REFERENCE

CONVENTIONS

** Page layout used for each command **

- Header

  Each command description starts on a new page with a header for quickly finding the command. This header indicates the command name and the two-character header used for the command syntax. Example:

  ```
  ============================================================
               DEFAULT SETUP                  DS
  ------------------------------------------------------------
  ```

  Where DEFAULT SETUP is the name of the command (no syntax),

  and DS are the first two characters used for the command syntax (not always the complete syntax).

- Purpose:
  Explains what the command does or what it is used for.

- Command Syntax:
  Shows the syntax for the command. Parameters are separated by commas. Commands are terminated by <cr> (carriage return).

- Response Syntax:
  Shows the format of the response from the ScopeMeter test tool. Responses are terminated by <cr> (carriage return). Each Response Syntax starts with the <acknowledge> response, followed by the query response if the syntax relates to a query.

- Example:
  This is an example QBASIC program which shows how you can use the command. The example may also include some other commands to show the relation with these commands.
  The following two comment lines (start with ') successively indicate the beginning and the end of an example program.

```
'*****************  Begin example program  ****************

'******************  End example program  *****************
```

Use an MS-DOS Editor and copy the complete program between these two lines to a file name with the .BAS extension.
Start QBASIC and open this file from the FILE menu.
Long programs (longer than 55 lines) include page breaks. Such page breaks are preceded by the ' (remark) character to prevent the QBASIC interpreter from interpreting them as an incorrect statement.
When you have connected the ScopeMeter test tool as indicated in the PM9080 Instruction Manual, you can start the program from the RUN menu.

** Syntax conventions **

The Command Syntax and the Response Syntax may contain the following characters:

UPPERCASE	These characters are part of the syntax. For commands, lower case is also allowed.
<...>	An expression between these brackets is a code, such as <cr> (carriage return) that can not be expressed in a printable character, or it is a parameter that is further specified. Do not insert the brackets in the command!
[...]	The item between these brackets is optional. This means that you may omit it. Do not insert the brackets in the command!
|	This is a separator between selectable items. This means that you must choose only one of the items (exclusive or).

** Overview of commands for the ScopeMeter test tool **

COMMAND NAME	COMMAND HEADER	PAGE NUMBER
CPL VERSION QUERY	CV	3.5
DEFAULT SETUP	DS	3.7
IDENTIFICATION	ID	3.9
PROGRAM COMMUNICATION	PC	3.11
QUERY PRINT	QP	3.16
RESET INSTRUMENT	RI	3.20
STATUS QUERY	ST	3.22
VIEW SCREEN	VS	3.25

CPL VERSION QUERY CV

Purpose:

Queries the CPL interface version.

Command Syntax:

CV<cr>

Response Syntax:

<acknowledge><cr><version><cr>

where
<version> is an ASCII string representing the year this version has been created.

Example:

```
'*****************  Begin example program  *****************

OPEN "COM1:1200,N,8,1,CS,DS,RB2048" FOR RANDOM AS #1
PRINT #1,"CV"             'Sends CPL VERSION query.
GOSUB Acknowledge         'Input acknowledge from test tool.
INPUT #1,VERSION$         'Inputs queried data.
PRINT "CPL Version "; VERSION$    'Displays version data.
END


'*****************  Acknowledge subroutine  ******************
'Use this subroutine after each command or query sent to the
'ScopeMeter test tool. This routine inputs the acknowledge
'response from the test tool. If the response is non-zero,
'the previous command was not correct or was not correctly
'received by the test tool. Then an error message is
'displayed and the program is aborted.

Acknowledge:
INPUT #1, ACK           'Reads acknowledge from test tool.
IF ACK <> 0 THEN
   PRINT "Error "; ACK; ": ";
   SELECT CASE ACK
     CASE 1
        PRINT "Syntax Error"
     CASE 2
        PRINT "Execution Error"
     CASE 3
        PRINT "Synchronization Error"
     CASE 4
        PRINT "Communication Error"
     CASE IS < 1
        PRINT "Unknown Acknowledge"
     CASE IS > 4
        PRINT "Unknown Acknowledge"
   END SELECT
   PRINT "Program aborted."
   END
END IF
RETURN

'******************  End example program  ******************
```

DEFAULT SETUP DS

Purpose:

Resets the ScopeMeter test tool to the factory settings at delivery, except for the RS232 communication settings such as baud rate, to keep the communication alive.
A Master Reset (refer to the Users Manual) performs the same, but also resets the RS232 communication settings to the default values.

Command Syntax:

DS<cr>

Response Syntax:

<acknowledge><cr>

Note: Wait for at least 2 seconds after the <acknowledge> reply has been received, to let the test tool settle itself before you send the next command.

Example:

```
'*****************  Begin example program  *****************

OPEN "COM1:1200,N,8,1,CS,DS,2048" FOR RANDOM AS #1
CLS
PRINT #1, "DS"          'Sends DEFAULT SETUP command.
GOSUB Acknowledge       'Input acknowledge from test tool.
SLEEP 2                 'Delay (2 s) necessary after "DS".
PRINT #1, "ID"          'Sends the IDENTIFICATION query.
GOSUB Acknowledge       'Input acknowledge from test tool.
INPUT #1, ID$           'Inputs identity data from test tool.
PRINT ID$               'Displays identity data.
CLOSE #1
END


'****************  Acknowledge subroutine  ******************
'Use this subroutine after each command or query sent to the
'ScopeMeter test tool. This routine inputs the acknowledge
'response from the test tool. If the response is non-zero,
'the previous command was not correct or was not correctly
'received by the test tool. Then an error message is
'displayed and the program is aborted.

Acknowledge:
INPUT #1, ACK           'Reads acknowledge from test tool.
IF ACK <> 0 THEN
   PRINT "Error "; ACK; ": ";
   SELECT CASE ACK
     CASE 1
        PRINT "Syntax Error"
     CASE 2
        PRINT "Execution Error"
     CASE 3
        PRINT "Synchronization Error"
     CASE 4
        PRINT "Communication Error"
     CASE IS < 1
        PRINT "Unknown Acknowledge"
     CASE IS > 4
        PRINT "Unknown Acknowledge"
   END SELECT
   PRINT "Program aborted."
   END
END IF
RETURN

'*****************  End example program  *****************
```

IDENTIFICATION ID

Purpose:

Returns information about the model number of the ScopeMeter test tool and its firmware version.

Command Syntax:

ID<cr>

Response Syntax:

<acknowledge><cr><identity><cr>

where,
<identity> is an ASCII string

Example:

The following example program queries the identity data of the ScopeMeter test tool and displays this data on the PC screen.

```
'*****************  Begin example program  *****************

CLS                      'Clears the PC screen.
OPEN "COM1:1200,N,8,1,CS,DS,RB2048" FOR RANDOM AS #1
PRINT #1, "ID"           'Sends IDENTIFICATION query.
GOSUB Acknowledge        'Input acknowledge from test tool.
INPUT #1, IDENT$         'Inputs the queried data.
PRINT IDENT$             'Displays queried data.
CLOSE #1
END


'*****************  Acknowledge subroutine  ******************
'Use this subroutine after each command or query sent to the
'ScopeMeter test tool. This routine inputs the acknowledge
'response from the test tool. If the response is non-zero,
'the previous command was not correct or was not correctly
'received by the test tool. Then an error message is
'displayed and the program is aborted.

Acknowledge:
INPUT #1, ACK            'Reads acknowledge from test tool.
IF ACK <> 0 THEN
   PRINT "Error "; ACK; ": ";
   SELECT CASE ACK
     CASE 1
        PRINT "Syntax Error"
     CASE 2
        PRINT "Execution Error"
     CASE 3
        PRINT "Synchronization Error"
     CASE 4
        PRINT "Communication Error"
     CASE IS < 1
        PRINT "Unknown Acknowledge"
     CASE IS > 4
        PRINT "Unknown Acknowledge"
   END SELECT
   PRINT "Program aborted."
   END
END IF
RETURN

'*****************  End example program  *****************
```

PROGRAM COMMUNICATION PC

Purpose:

Programs the following settings used for RS232 communication:

1. Baud rate
2. Parity
3. Number of data bits
4. Number of stop bits
5. Handshake method

Command Syntax:

PC <baudrate>,<parity>,<data bits>,<stop bits>
[,<handshake>]

where,

<baudrate>	=	75|110|150|300|600|1200|2400|4800|9600|19200|38400
<parity>	=	O|E|N (Odd, Even, or No parity)
<data bits>	=	7|8
<stop bits>	=	1
<handshake>	=	XONXOFF (software handshaking)

Default settings after power-on:

<baudrate>	=	1200
<parity>	=	N
<data bits>	=	8
<stop bits>	=	1
<handshake>	=	disabled (see the following notes)

Notes:

Option XONXOFF takes care off handshaking to prevent overflow of the receive buffer (length 254 bytes). The test tool sends XOFF (ASCII code 19 decimal) to tell the computer that it must postpone data transmission until it receives XON (ASCII code 17 decimal). This protocol will only work if the computer reacts on XON and XOFF codes. It is recommended to set the read buffer length for the computer to 2048 or higher as shown in several example programs, with the following QBASIC command:

```
OPEN "COM1:1200,N,8,1,CS,DS,RB2048" FOR RANDOM AS #1
```

When you use PROGRAM WAVEFORM (PW) or QUERY WAVEFORM (QW) commands, you MUST omit option XONXOFF. The PW and QW commands use binary data that may contain XON and XOFF codes.

Response Syntax:

<acknowledge><cr>

Example:

The following program assumes that the ScopeMeter test tool is connected to the COM1 port of the PC and that the power-on defaults are valid. The program switches over to baud rate 19200. This includes first programming the test tool and then the RS232 port of the PC to baud rate 19200. This higher baud rate considerably increases the data transfer speed. The program then sends the QP query to read the test tool screen (print) data and copies this data to the printer port LPT1. The print format is suitable for an EPSON FX/LQ compatible printer.
At the end of the program the baud rate is set back to 1200 (default baud rate).

```
'*****************  Begin example program  *****************

CLS                          'Clears the PC screen.
OPEN "COM1:1200,N,8,1,CS,DS,RB2048" FOR RANDOM AS #1
                             'Programs COM1 port parameters to
                             'match with the ScopeMeter power-on
                             'defaults.
PRINT #1, "PC19200,N,8,1" 'Programs ScopeMeter to a higher
                             'baud rate.
GOSUB Acknowledge            'Input acknowledge from test tool.
CLOSE #1
OPEN "COM1:19200,N,8,1,CS,DS,RB7500" FOR RANDOM AS #1
                             'Programs COM1 port parameters to
                             'match with the new ScopeMeter
                             'settings.
PRINT #1, "QP"               'Sends QUERY PRINT data command.
GOSUB Acknowledge            'Input acknowledge from test tool.
INPUT #1, Count              'Number of bytes that will follow.
PRINT STR$(Count+1)+" Bytes will be input from test tool."
PRINT
PRINT "Busy reading print data !"
PRINT
GOSUB Response
PRINT #1, "PC1200,N,8,1"  'Programs ScopeMeter back to the
                             'default communication settings.
GOSUB Acknowledge            'Input acknowledge from test tool.

PRINT STR$(LEN(Resp$))+" Bytes have been input from test tool"
PRINT
PRINT "The test tool screen image is now copied to the"
PRINT "printer connected to LPT1."
OPEN "LPT1" FOR OUTPUT AS #2
PRINT #2, LEFT$(Resp$, Count)   'Copies the image data to the
                                'printer on LPT1.
CLOSE                           'Closes all files.
END
'
```

```
'***************  Acknowledge subroutine  *****************
'Use this subroutine after each command or query sent to the
'ScopeMeter test tool. This routine inputs the acknowledge
'response from the test tool. If the response is non-zero,
'the previous command was not correct or was not correctly
'received by the test tool. Then an error message is
'displayed and the program is aborted.

Acknowledge:
INPUT #1, ACK           'Reads acknowledge from test tool.
IF ACK <> 0 THEN
   PRINT "Error "; ACK; ": ";
   SELECT CASE ACK
     CASE 1
        PRINT "Syntax Error"
     CASE 2
        PRINT "Execution Error"
     CASE 3
        PRINT "Synchronization Error"
     CASE 4
        PRINT "Communication Error"
     CASE IS < 1
        PRINT "Unknown Acknowledge"
     CASE IS > 4
        PRINT "Unknown Acknowledge"
   END SELECT
   PRINT "Program aborted."
   END
END IF
RETURN
'
```

```
'****************** Response subroutine *********************
'This subroutine reads bytes from the RS232 buffer as long
'as they enter. When no bytes enter for 1 second, the program
'assumes that the test tool has terminated its response.
'All bytes that enter the buffer are appended to the string
'Resp$.

Response:
  start! = TIMER
  'Wait for bytes (maximum 1 s) to enter RS232 buffer.
  WHILE ((TIMER < (start! + 1)) AND (LOC(1) = 0))
  WEND
  IF LOC(1) > 0 THEN       'If RS232 buffer contains bytes.
       Resp$ = ""
       DO
           ' LOC(1) gives the number of bytes waiting:
           ScopeInput$ = INPUT$(LOC(1), #1)    'Input bytes
           Resp$ = Resp$ + ScopeInput$         'Append bytes
           start! = TIMER
           WHILE ((TIMER < (start! + 1)) AND (LOC(1) = 0))
           WEND
       LOOP WHILE LOC(1) > 0   'Repeat as long as bytes enter.
  END IF
RETURN

'******************  End example program  ******************
```

QUERY PRINT QP

Purpose:

Queries the print data of the ScopeMeter test tool in EPSON FX/LQ format, binary coded. This allows you to make a copy of the test tool screen on paper. Also see the VS (VIEW SCREEN) command.

Command Syntax:

QP<cr>

Response Syntax:

<acknowledge><cr>

If <acknowledge> value 0 is returned, this response is immediately continued with the following data:

<count>,<print data><checksum>

where,

<count>	=	Decimal number of bytes in the <print data> block.
<print data>	=	Binary data bytes. The decimal value of each byte is in the range from 0 to 255.
<checksum>	=	One-byte checksum over all bytes in <print data>.

Example:

The following program reads the test tool screen (print) data and copies this data to the printer port LPT1.
The print format is suitable for an EPSON FX/LQ compatible printer.
The data transfer speed (baud rate) is set to 19200 and after the output it is set back to 1200 (default baud rate).

```
'*****************  Begin example program  *****************

CLS
OPEN "COM1:1200,N,8,1,CS,DS,RB2048" FOR RANDOM AS #1
                          'Programs COM1 port parameters to
                          'match with the ScopeMeter power-on
                          'defaults.
PRINT #1, "PC19200,N,8,1" 'Programs ScopeMeter to a higher
                          'baud rate.
GOSUB Acknowledge         'Input acknowledge from test tool.
CLOSE #1
OPEN "COM1:19200,N,8,1,CS,DS,RB2048" FOR RANDOM AS #1
                          'Programs COM1 port parameters to
                          'match with the new ScopeMeter
                          'settings.
PRINT #1, "QP"            'Sends QUERY PRINT data command.
GOSUB Acknowledge         'Input acknowledge from test tool.
INPUT #1, Count           'Number of bytes that will follow.
PRINT STR$(Count+1)+" Bytes will be input from test tool."
PRINT
PRINT "Busy reading print data !"
PRINT
GOSUB Response
PRINT #1, "PC1200,N,8,1"  'Programs ScopeMeter back to the
                          'default communication settings.
GOSUB Acknowledge         'Input acknowledge from test tool.

PRINT STR$(LEN(Resp$))+" Bytes have been input from test tool"
PRINT
PRINT "The test tool screen image is now copied to the"
PRINT "printer connected to LPT1."
OPEN "LPT1" FOR OUTPUT AS #2
PRINT #2, LEFT$(Resp$, Count)   'Copies the image data to the
                                'printer on LPT1.
CLOSE                           'Closes all files.
END
'
```

```
'***************  Acknowledge subroutine  *****************
'Use this subroutine after each command or query sent to the
'ScopeMeter test tool. This routine inputs the acknowledge
'response from the test tool. If the response is non-zero,
'the previous command was not correct or was not correctly
'received by the test tool. Then an error message is
'displayed and the program is aborted.

Acknowledge:
INPUT #1, ACK           'Reads acknowledge from test tool.
IF ACK <> 0 THEN
   PRINT "Error "; ACK; ": ";
   SELECT CASE ACK
     CASE 1
        PRINT "Syntax Error"
     CASE 2
        PRINT "Execution Error"
     CASE 3
        PRINT "Synchronization Error"
     CASE 4
        PRINT "Communication Error"
     CASE IS < 1
        PRINT "Unknown Acknowledge"
     CASE IS > 4
        PRINT "Unknown Acknowledge"
   END SELECT
   PRINT "Program aborted."
   END
END IF
RETURN
'
```

```
'****************** Response subroutine *********************
'This subroutine reads bytes from the RS232 buffer as long
'as they enter. When no bytes enter for 1 second, the program
'assumes that the test tool has terminated its response.
'All bytes that enter the buffer are appended to the string
'Resp$.

Response:
  start! = TIMER
  'Wait for bytes (maximum 1 s) to enter RS232 buffer.
  WHILE ((TIMER < (start! + 1)) AND (LOC(1) = 0))
  WEND
  IF LOC(1) > 0 THEN       'If RS232 buffer contains bytes.
       Resp$ = ""
       DO
           ' LOC(1) gives the number of bytes waiting:
           ScopeInput$ = INPUT$(LOC(1), #1)    'Input bytes
           Resp$ = Resp$ + ScopeInput$         'Append bytes
           start! = TIMER
           WHILE ((TIMER < (start! + 1)) AND (LOC(1) = 0))
           WEND
       LOOP WHILE LOC(1) > 0   'Repeat as long as bytes enter.
  END IF
RETURN

'******************  End example program  ******************
```

RESET INSTRUMENT RI

Purpose:

Performs the following actions:
- Resets the firmware of the ScopeMeter test tool.
- Clears input and output buffers.

The RI command does not affect the following:
- Waveform, screen, and setup memories.
- Mode of operation (Scope, Meter, or Component Tester).
- RS232 communication settings (e.g. baud rate), to keep the communication alive.

Command Syntax:

RI<cr>

Response Syntax:

<acknowledge><cr>

Note: Wait for at least 2 seconds after the <acknowledge> reply has been received, to let the test tool settle itself before you send the next command.

Example:

The following example resets the test tool and waits for 2 seconds to let the test tool execute the reset and become ready for next commands.
The test tool is queried for the identification data; this data is input and displayed on the PC screen.

```
'*****************  Begin example program  *****************

CLS                      'Clears the PC screen.
OPEN "COM1:1200,N,8,1,CS,DS,RB2048" FOR RANDOM AS #1
PRINT #1, "RI"           'Sends the RESET INSTRUMENT command.
GOSUB Acknowledge        'Input acknowledge from test tool.
SLEEP 2                  'Delay (2 s) necessary after reset.
PRINT #1, "ID"           'Sends IDENTIFICATION query.
GOSUB Acknowledge        'Input acknowledge from test tool.
INPUT #1, IDENT$         'Inputs the queried data.
PRINT IDENT$             'Displays queried data.
CLOSE #1
END


'****************  Acknowledge subroutine  ******************
'Use this subroutine after each command or query sent to the
'ScopeMeter test tool. This routine inputs the acknowledge
'response from the test tool. If the response is non-zero,
'the previous command was not correct or was not correctly
'received by the test tool. Then an error message is
'displayed and the program is aborted.

Acknowledge:
INPUT #1, ACK            'Reads acknowledge from test tool.
IF ACK <> 0 THEN
   PRINT "Error "; ACK; ": ";
   SELECT CASE ACK
     CASE 1
        PRINT "Syntax Error"
     CASE 2
        PRINT "Execution Error"
     CASE 3
        PRINT "Synchronization Error"
     CASE 4
        PRINT "Communication Error"
     CASE IS < 1
        PRINT "Unknown Acknowledge"
     CASE IS > 4
        PRINT "Unknown Acknowledge"
   END SELECT
   PRINT "Program aborted."
   END
END IF
RETURN

'*****************  End example program  *****************
```

STATUS QUERY ST

Purpose:

Queries the error status of the ScopeMeter test tool. This is a 16-bit word, presented as an integer value, where each bit represents the Boolean value of a related error event. After the reply or after a RI (Reset Instrument) command, the value is reset to zero.
A complete description of the status word is given in Appendix B.

Command Syntax:

ST<cr>

Response Syntax:

<acknowledge><cr>

If <acknowledge> value 0 is returned, this response is immediately continued with the following data:

<status><cr>

where,
<status> = integer value 0 to 32767

Example:

The following example program sends a wrong command to the test tool to test the Acknowledge subroutine and to check the status returned from the ST query.
The acknowledge subroutine contains a GOSUB Status.display to input the status data from the test tool when the acknowledge response is non-zero (ACK <> 0).

```
'*****************  Begin example program  *****************

CLS                          'Clears the PC screen.
OPEN "COM1:1200,N,8,1,CS,DS,RB2048" FOR RANDOM AS #1
PRINT #1, "PC12345,N,8,1"  'Sends a baud rate value that is
                         '   out of range for the test tool.
GOSUB Acknowledge.Status   'Input acknowledge from test tool
                           'and the status value if the
                           'acknowledge value is non-zero.
END


'*************  Acknowledge + Status subroutine  ***********
'This subroutine inputs the acknowledge value from the
'ScopeMeter test tool. If the acknowledge value is non-zero,
'the ST query is used to get further status information from
'the test tool with respect to the error.
'In case of an error the program is aborted.

Acknowledge.Status:
INPUT #1, ACK           'Reads acknowledge from test tool.
IF ACK <> 0 THEN
   PRINT "Error "; ACK; ": ";
   SELECT CASE ACK
     CASE 1
        PRINT "Syntax Error"
     CASE 2
        PRINT "Execution Error"
     CASE 3
        PRINT "Synchronization Error"
     CASE 4
        PRINT "Communication Error"
     CASE IS < 1
        PRINT "Unknown Acknowledge"
     CASE IS > 4
        PRINT "Unknown Acknowledge"
   END SELECT
   GOSUB Status.display         'Further specifies the error.
   PRINT "Program aborted."
   END
END IF
RETURN
'
```

```
'**************  Displays test tool status *****************

'This subroutine gives you further information if the
'acknowledge reply from the ScopeMeter test tool is non-zero.

Status.display:
PRINT #1, "ST"             'Sends the STATUS query.
GOSUB Acknowledge.Status   'Inputs acknowledge from test tool.
INPUT #1, STAT             'Inputs status value.
PRINT "Status " + STR$(STAT) + ": ";
SELECT CASE STAT
  CASE 0
     PRINT "No error"
  CASE 1
     PRINT "Unknown header"
  CASE 2
     PRINT "Data format of body is wrong"
  CASE 4
     PRINT "Data out of range"
  CASE 8
     PRINT "Invalid instruction in present mode"
  CASE 16
     PRINT "Called function not implemented"
  CASE 32
     PRINT "Invalid number of parameters"
  CASE 64
     PRINT "Wrong number of data bits"
  CASE 512
     PRINT "Conflicting instrument settings"
  CASE 16384
     PRINT "Checksum error"
END SELECT
RETURN

'******************  End example program  ******************
```

VIEW SCREEN VS

Purpose:

Restores a screen saved by keypad operation on the test tool display. You can use this command in combination with the QP (Query Print) command to transfer saved screens to a computer. The ScopeMeter test tool Users Manual gives information on how to save screens.

Command Syntax:

VS <view screen><cr>

where,
<view screen> = 0 to 5

0	Exit View Screen mode
1	View Screen 1
2	View Screen 2
3	View Screen 3
4	View Screen 4
5	View Screen 5

The following commands will cause View Screen mode to exit: RI, DS, and RS.
The QP command will always return the data related to the actual screen.

Command Syntax:

<acknowledge><cr>

Example:

The following example program recalls the screen saved previously in screen memory 5.

```
'*****************  Begin example program  *****************

CLS                      'Clears the PC screen
OPEN "COM1:1200,N,8,1,CS,DS,RB2048" FOR RANDOM AS #1
PRINT #1,"VS 5"           'Sends VIEW SCREEN 5 command.
GOSUB Acknowledge         'Input acknowledge from test tool.
PRINT "The waveform from VIEW SCREEN 5 is now displayed."
PRINT
PRINT "You can use the example program for the QP command"
PRINT "to copy the screen image to an EPSON FX/LQ compatible"
PRINT "printer."
END


'*****************  Acknowledge subroutine  ******************
'Use this subroutine after each command or query sent to the
'ScopeMeter test tool. This routine inputs the acknowledge
'response from the test tool. If the response is non-zero,
'the previous command was not correct or was not correctly
'received by the test tool. Then an error message is
'displayed and the program is aborted.

Acknowledge:
INPUT #1, ACK            'Reads acknowledge from test tool.
IF ACK <> 0 THEN
   PRINT "Error "; ACK; ": ";
   SELECT CASE ACK
     CASE 1
        PRINT "Syntax Error"
     CASE 2
        PRINT "Execution Error"
     CASE 3
        PRINT "Synchronization Error"
     CASE 4
        PRINT "Communication Error"
     CASE IS < 1
        PRINT "Unknown Acknowledge"
     CASE IS > 4
        PRINT "Unknown Acknowledge"
   END SELECT
   PRINT "Program aborted."
   END
END IF
RETURN

'******************  End example program  ******************
```

APPENDIX A ACKNOWLEDGE DATA

The ScopeMeter test tool returns an <acknowledge> reply after each command or query. The value indicates correct or incorrect operation. You always must read this reply to check for the correct operation and to achieve synchronization between your program and the RS232 interface of the ScopeMeter test tool.

<acknowledge>

VALUE	MEANING
0	No Error
1	Syntax Error (see Note)
2	Execution Error (see Note)
3	Synchronization Error
4	Communication Error

Note: The ST query may give you additional information.

When the ScopeMeter test tool detects an error during the execution of a command, it sends the corresponding <acknowledge> reply, terminates further execution of the command and will be ready to accept a new command.

Syntax Error

Returned when the command is not understood by the ScopeMeter test tool for one of the following reasons :

- Unknown header
- Wrong instructions
- Data format of body is wrong, e.g. alpha characters when decimal data is needed.

Execution Error

Returned when internal processing is not possible because of one of the following reasons:

- Data out of range
- Conflicting instrument settings

Synchronization Error

Returned when the ScopeMeter test tool receives data while it does not expect any data. This can occur as follows:

- The ScopeMeter test tool receives a new command while a previous command or query is not yet completely executed. You can prevent this error by doing the following:

  1. Read the <acknowledge> reply after each command or query.
  2. If this <acknowledge> is zero and if a query was sent to the ScopeMeter test tool, read all available response data.

Communication Error

Any framing, parity or overrun error detected on the received data will cause Communication Error.

APPENDIX B STATUS DATA

The Status word returned from the ST query gives you extra information when you have received a non-zero <acknowledge> reply.
The Status word is a 16-bit binary word where each bit set true represents an error event with a decimal value determined by the bit position. (See the following table.)

When more than one bit is set true in the status word, the response from the ST query will be the sum of the decimal values of the individual bits.

Example:

<status> = 34 This equals 32 + 2
2 = Wrong parameter data format
32 = Invalid number of parameters

BIT	DECIMAL VALUE	EVENT DESCRIPTION	<acknowledge> VALUE
0	1	Illegal command	1
1	2	Wrong parameter data format	1
2	4	Parameter out of range	1 or 2
3	8	Instruction not valid in present state	1
4	16	Called function not implemented	2
5	32	Invalid number of parameters	2
6	64	Wrong number of data bits	2
9	512	Conflicting instrument settings	2
14	16384	Checksum error	2

Remarks:

1. A bit in the status word is set when the corresponding error event occurs.
2. Bits do not affect each other.
3. New error events will 'accumulate' in the status word. This means existing bits remain set.

The status word is cleared (all bits reset) as follows:

1. After the response (the status word) from the ST query has been read.
2. After the RI (Reset Instrument) command.