



Programming MACH JTAG-ISP Devices on Automated Test Equipment

Application Note

Programming MACH JTAG-ISP Devices on Automated Test Equipment



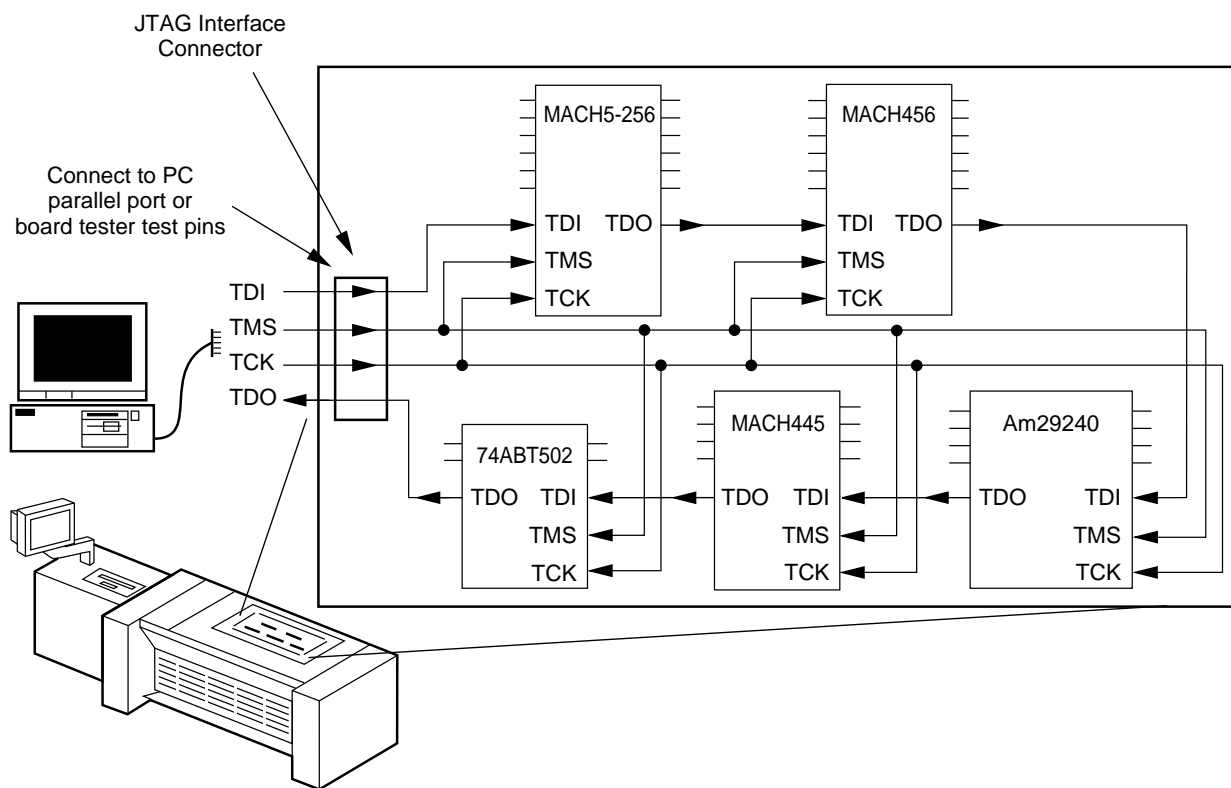
Application Note (MACH/PLD)

by Arthur Khu

INTRODUCTION

MACH CPLDs with IEEE 1149.1 (JTAG) test circuitry are in-system programmable through the test access port pins by a PC or Automated Test Equipment (ATE). This offers advantages in the design, manufacturing, and maintenance phases of a product's life cycle.

Designers can develop systems with reconfigurable MACH JTAG CPLDs connected in series in a boundary scan chain with other JTAG-compliant devices for testability (Figure 1).



21146A-1

Figure 1. Board with 5 JTAG Devices Connected in a Serial JTAG Chain. TDI and TDO are Connected in Series While TMS and TCK are in Parallel

JTAG-ISP: A LONG-TERM, COST-EFFECTIVE SOLUTION

You can use an AMD-developed PC-based software tool called MACHPRO® to configure the MACH device on the board through the same 1149.1 test interface used for board testing via a PC parallel port programming cable. Any subsequent logic changes in the MACH device can be quickly performed on the board from your PC development station without having to remove, reprogram, or reinsert the parts. You can therefore attempt more design iterations to debug and improve product performance.

Once the design has been finalized and ready for manufacturing, MACHPRO can generate an output file for the target ATE or board tester. You can now incorporate the programming of the MACH CPLDs into the board manufacturing flow. Programming MACH devices on the tester during manufacturing offers many advantages:

- Programming and pattern verification is fast and is determined by the tester clock rate and the programming time requirements of the CPLD
- Programming on the ATE removes the cost of maintaining and upgrading a separate CPLD programming station
- Component damage is minimized by reducing handling of CPLDs with fine-pitch leads
- Planning and inventory charges for MACH CPLDs are reduced

MACH CPLDs can be treated as generic devices which can be loaded directly on a printed-circuit board and configured with the required patterns during the manufacturing and test flow. Programming on the tester eliminates the possibility of a CPLD with the wrong pattern being placed on the board. Since MACH devices have fast programming times, combining programming and board test on the same ATE station will have minimal impact on the manufacturing beat rate.

Once the product has been released to the end-user, logic updates can still be easily performed in the field by a technical support person with the new programming patterns, a notebook PC, and a programming cable. You can also design the board such that MACH

devices in the JTAG chain can be configurable through a microcontroller. Update software and the new programming patterns can be delivered via disk or modem to customers with these microcontroller- or microprocessor-based systems and they can perform the updates themselves.

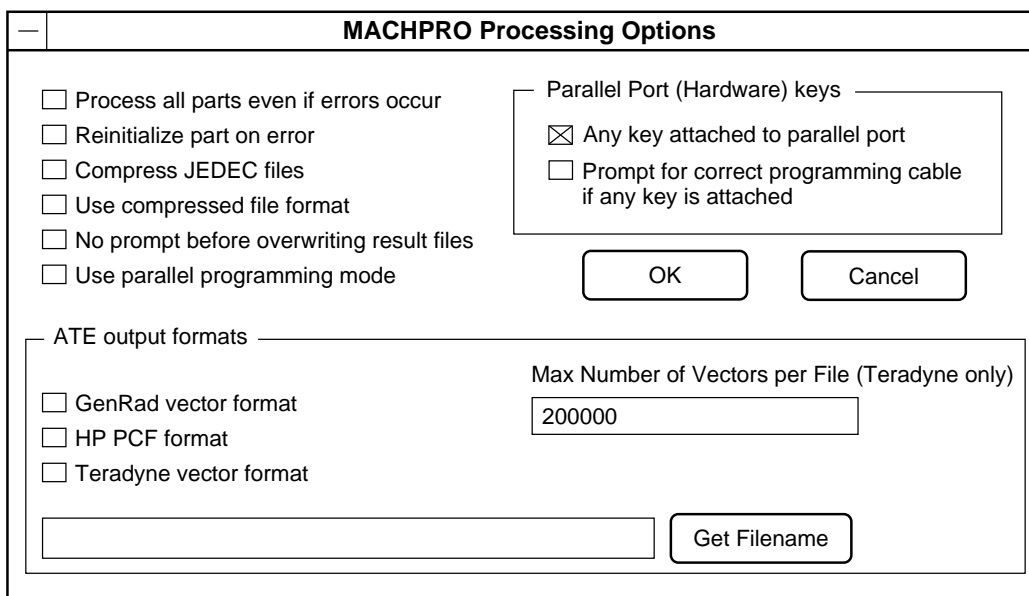
Programming on Board Testers

MACHPRO has options for generating vector files for programming MACH CPLDs on board testers from the major ATE vendors: GenRad, HP, and Teradyne. Since MACH JTAG CPLDs are 1149.1-compliant, MACHPRO can generate programming and pattern verification vectors for MACHs that are in a serial chain with other non-AMD non-MACH JTAG-compliant devices. The position of the MACH devices in the serial chain and the operations to be performed are described in an ASCII file called a JTAG chain description file. Refer to the *MACHPRO User Guide (PID# 18905C)* for information on creating and processing a JTAG chain description file). Any MACH parts that do not need to be reconfigured and all non-MACH devices are put into bypass mode by MACHPRO.

By using the 1149.1/JTAG interface for MACH CPLD programming, the manufacturing engineer does not need any special programming software because the programming vector files can be treated as regular test programs. The manufacturing engineer can therefore process the MACHPRO-generated files with existing JTAG or boundary scan test software supplied by the ATE vendor, and convert it to the native tester language format before downloading it to the tester. Refer to the AMD application note (*PID# 20985A*) *In-System Programming on an HP3070 through the MACH CPLDs IEEE 1149.1 Test Access Port* for general board design and test fixture setup guidelines to facilitate on-board programming.

Generating Vector Files

There are Windows 3.1, Win95, and DOS versions of MACHPRO. To generate the vector files in the Windows versions, click on the desired output format options (Figure 2). There are equivalent command line options for generating ATE vector files in the DOS version of MACHPRO.



21146A-2

Figure 2. MACHPRO for Windows Output Option Menu for Specifying ATE Vector Formats

Generating a GenRad Vector File

To generate a vector file for a GenRad tester, use the “-4 filename” option.

```
Ex: C:\MPRO_DSG> machpro -I project -4
    board1.vct
```

MACHPRO will generate a vector file called BOARD1.VCT with vectors in the following format:

```
! File [board1.vct] created Wed May 22
  18:09:29 1996
! for GenRad preprocessor generated by
! MACHPRO(tm)
! Version 1.35g (c) 1994-1996 Advanced
  Micro Devices, Inc.
! Pin order: TCK,TMS,TDI,TRST,ENABLE,TDO
+unit Program_AMD_MACHS
00011X
00011X
C1011X
C1011X
+begin id 1 register
C1011H
C1011L
+end id 1
C1011X
+wait 50m
C1111X
C0111X
```

```
+verify begin id 1
C1011X
C0011X
...
C1111H
+end verify id 1
...
C0011X
00011X
+end unit
```

Comments are preceded by a ‘!’ and continue to the end of the line. Lines which require special processing are marked by a ‘+’. For example: The lines “+begin id 1 register” and “end id 1” are used to bracket vectors for testing the device ID code for the first JTAG device in the chain. If you have more than one MACH JTAG device being programmed in the chain, then you will have similar sets of vector statements for each device with the number being the position of the device in the chain.

Similarly, the statements “+verify begin id 1” and “+end verify id 1” are used to verify if JTAG device number 1 was configured correctly. There are no special statements to mark which parts are being programmed, but programming is performed by shifting in data and then waiting for a predetermined amount of time. This is accomplished for the MACH device by the line “+wait 50m” which means wait in this state for 50 ms.

Each line of the form XXXXXX (e.g., 00011H) specifies the state of each JTAG pin listed in the "Pin order" statement. In the preceding example, 00011H means drive TCK, TMS, and TDI low, drive TRST and ENABLE high, and test if the TDO pin is high. The tester drives the pins low or high if 0 or 1, respectively, are specified, and compares the state of the TDO pin at this time against H or L as indicated. If it is X, then the tester does not need to test the TDO pin. A "C" will be compiled into a clock pulse. The clock pulse will be issued only after the other inputs (i.e., TMS, TDI, TRST, and ENABLE) have been setup.

This vector file is processed by a GenRad-supplied program called AMD2GR.PRL to produce a ".DTS" file which is a GenRad intermediate file format. The .DTS file is a test program written as a model that can be stored in a library on the GenRad test system.

GenRad's test generator program is then run on the .DTS file to convert the model into a ".TPG" (test program) compatible format. The .TPG file is then processed further and converted into a binary file with the

".OBC" file extension. The .OBC file can now be downloaded and run on the tester.

GenRad offers a hardware option for their testers called Deep Serial Memory which eliminates the overhead in loading test vectors. This reduces total programming and pattern verification time and results in better tester throughput. Contact your local GenRad Applications Engineer for more details.

Generating an HP3070 Pattern Capture Format (PCF) File

PCF is the native tester language of the HP3070 series of testers. To generate a PCF file, use the "-2 PCF_file" command option in MACHPRO:

```
Ex: C:\MPRO_DSG> machpro -I project -2
    design1.pcf
```

MACHPRO will generate a PCF file called DESIGN1.PCF. The file format is very similar to the GenRad format:

```
! Thu Jun 20 15:06:25 1996
! HP PCF File [design1.pcf] generated by MACHPRO(tm)
! Version 1.35g (c) 1994-1996 Advanced Micro Devices, Inc.
! PCF header by APG Test Consultants
! Pattern Capture Format      subset of VCL
! Vector Control Language    digital test language

!generate static test

vector cycle 500n
receive delay 400n

family TTL

assign TCK      to nodes "TCK_Node"      ! Enter nodes
assign TMS      to nodes "TMS_Node"
assign TDI      to nodes "TDI_Node"
assign TDO      to nodes "TDO_Node"
assign TRST     to nodes "TRST_Node"
assign PROGRAM  to nodes "PROGRAM_PIN_NODE"

inputs  TCK,TMS,TDI,TRST,PROGRAM
outputs TDO

!dynamic TCK,TMS,TDI,TDO

pcf order is TCK,TMS,TDI,TRST,PROGRAM,TDO

unit "Program_AMD_MACHS"

pcf
!
!Start of vectors
"00011X"
"00011X"      ! Logic Rst
"01011X"
```

```

"11011X"      ! Logic Rst
"01011X"
"11011X"      ! Logic Rst
"11011X"      ! Update IR
"00011X"
...
"10011X"      ! Test Idle
"00010X"      ! Test Idle
end pcf
wait 100m
pcf
"00011X"      ! Test Idle
!== Prog Init/shift row all 0s
!== Shift in instruction 3
"01011X"
"11011X"      ! Select DR
"01011X"
"11011X"      ! Select IR
end pcf
end unit

```

A comment is preceded by the exclamation point “!” and continues to the end of the line. The `generate static test` and `dynamic` statements are valid HP3070 syntax but are commented out in the MACHPRO-generated programming files. These features are not being used at this time.

The vector cycle and receive delay times indicate the application rate of the vectors and when the receive strobe is activated.

```

vector cycle 500n
receive delay 400n

```

In the preceding example an individual PCF vector is applied every 500 nano seconds. The vector is driven by the tester for 500 nano seconds. If any responses are to be measured (out of TDO) by the tester it will be measured 400 nano seconds after driving the inputs. The programming vectors produced by MACHPRO use a 50% duty cycle on TCK. Therefore, one cycle of TCK high (1) and low (0) will be represented by two PCF vectors lasting 500 n seconds each, translating into a TCK rate of 1 MHz.

The assignment statements map the test pin variable names (TCK, TMS, TDO, etc.) to the signal names in the board and fixture file. The input and output sections are used to assign test pins to a pin driver or receiver

on the tester. The order of signals in a vector is determined by the PCF order statement and is similar to the GenRad format. There are comments included to the right of some vectors to indicate the state of the JTAG state machine when the vector is executed. PCF also has a `wait` statement to hold the tester drive pins in the current vector state for the specified amount of time.

The PCF file must be compiled first before downloading it to a tester. Transfer the PCF file from the PC to the HP3070 and then do the following:

1. Change the file type from a text file to a digital file by loading the PCF file in an HP BASIC window, running the command `load digital "design1.pcf"`, and then re-saving it.
2. Assign node names to the PCF file.

Each programming file generated by MACHPRO contains dummy node names (place holders) called `TCK_NODE`, `TMS_NODE`, `TDI_NODE`, `TDO_NODE`, `TRST_NODE`, and `PROGRAM_PIN_NODE`. After loading the programming file, edit the node names in the assign statements to match those found in the board file. The following shows an example of a PCF programming file where the actual node names on the board are: `TCK`, `TMS`, `TDI`, `TDO_TDI_2`, `N_BSCAN_RST`, `N$3865`.

```

assign TCK      to nodes "TCK"  ! Enter nodes
assign TMS      to nodes "TMS"
assign TDI      to nodes "TDI"
assign TDO      to nodes "TDO_TDI_2"
assign TRST     to nodes "N_BSCAN_RST"
assign PROGRAM  to nodes "N$3865"

```

If the PROGRAM node or the TRST node are to be controlled by general purpose (GP) relays or by other persistent methods (i. e., pull-ups or pull-downs), then

use an asterisk (*) so the tester will not drive these nodes when executing the test program.

```

assign TCK      to nodes "TCK"  ! Enter nodes
assign TMS      to nodes "TMS"
assign TDI      to nodes "TDI"
assign TDO      to nodes "TDO_TDI_2"
assign TRST     to nodes *
assign PROGRAM  to nodes *

```

The TRST and PROGRAM nodes should have an asterisk only if they are being controlled by GP relays. These signals should not glitch during programming and we therefore recommend controlling these signals directly with GP relays at all times during JTAG-ISP.

The MACHPRO-generated PCF files are large and it is necessary to partition this file into smaller files. AMD provides an "Automatic Partition Generator" (APG) program that runs on the HP3070 platform. APG will automatically partition files into sizes that the tester can handle. APG will break or partition files at a point where the critical signals TCK and TMS are being driven high. The last vector in one file is always the first vector in the next file and helps to maintain glitch-free program transition. If the test programmer has adequately taken care of the persistence of critical signals along with any disabling required, the tests should transition glitch-free.

To use the APG program, copy the file to your HP3070, open a BASIC window in the directory where your MACHPRO files reside, and then load the APG BASIC program. Run it using the following commands:

1. Type `load basic "apg.bas"` at the prompt. The APG code should now be loaded and visible in the window.
2. When APG.BAS is loaded, type `run`.
3. The program will prompt you for a filename. Enter the MACHPRO-generated PCF filename you created and press return.

```

----- APG Test Consultants, Inc.-----
----- MACH IEEE.1149.1 Programming----
----- PCF File Partitioning Utility ----
-----
Enter the PCF MACH file you would like to
Partition >

```

The actual number of partitioned test files that the APG utility will generate depends on the size of the original PCF file. As a rule of thumb one file partition is created for each device being programmed. If you have a JTAG chain with two MACH445 devices, then two partitions will be generated. For a single MACH4xx device not in a large chain it is possible that the MACHPRO-generated PCF test file will compile without partitioning. Try to compile the MACHPRO-generated PCF test file first, and use the APG utility only if memory warning/error messages are generated by the PCF compiler.

The APG utility creates names for the partitioned files by appending an underscore and a number to the original file name to indicate the number of generated test files and the order in which to apply the tests. If the input file name was `mach-prog.pcf` then the files produced by the APG utility will be called `mach-prog.pcf_1`, `mach-prog.pcf_2`, `mach-prog.pcf_3`, etc.

The following is an example of an HP3070 test plan showing how to execute the tests in sequential order. The GP relay connect statements are there to insure the persistence of the critical signals during programming.

```

gpconnect "disabling_nodes" to "VCC"
gpconnect "TRST"            to "VCC"
gpconnect "PROG"            to "GROUND"

test "digital/mach-prog.pcf_1"
test "digital/mach-prog.pcf_2"
test "digital/mach-prog.pcf_3"

```

After the files are partitioned they will have to be added to the HP3070 test order file and compiled. The files can take up to one hour to compile depending upon the system load. When the files are compiled they are ready for execution on the tester.

PROGRAMMING ON TERADYNE TESTERS

There are two ways to program a MACH device on the Z18XX series testers: through the Vector Processor (VP) or the Digital Function Processor (DFP). The VP takes MACHPRO-generated programming vectors and applies them to the JTAG interface while the DFP programs a MACH by processing the programming information specified in a JEDEC map.

The Teradyne Vector Processor

Use the MACHPRO command line with the “-3 file-name maxvect” option to generate the vectors for the VP:

```
Ex: C:\> machpro -I design3.chn -z 3 -3
      teradyne.vct 150000
```

The maxvect parameter is used to specify the maximum number of vectors in a vector file. If this number is exceeded, then MACHPRO automatically creates a new file or files to handle the overflow. The files will have the names 0000000n.AMD where n ranges from 1 to the maximum number of files required.

Upon completion of vector generation, MACHPRO will display a message indicating the total number of vectors generated and the number of new files created:

```
C:\JTAG> machpro -i design3.chn -z 3 -3 teradyne.vct 150000
MACHPRO(tm) Version 1.35g (c) 1994-1996 Advanced Micro Devices, Inc.
[Start: Fri Jun 21 17:42:31 1996]
=====
[          board_00 (  mach445)]: Program, plus pattern verification only
Reading JEDEC map [blink.jed]
Reading row      [          0]
==> Teradyne vectors written to file [teradyne.vct]
==> No errors
=====
[End   : Fri Jun 21 17:46:49 1996]
Elapsed time      (00:04:18)

Number of vectors generated in all files [399959]
Number of additional tester files created [2]
C:\JTAG>
```

The vector files have to be partitioned to prevent overflowing the VP memory. A local library is created in the VP system and the vectors are loaded into this library.

For example: If maxvect is specified as 150000 and the total number of vectors to perform the programming operation is 400 K, then MACHPRO will create 2 additional files with the names 00000001.AMD and 00000002.AMD. The vectors will be partitioned at the

point where TCK is high. This last vector will be the first vector in the new vector file.

The Teradyne file format for the VP contains pin order declarations, tester clock frequency, and vectors (Figure 3). Inputs in the vector are represented by H (High/1) and L (Low/0) while outputs are represented by U (Up/HIGH/1), D (Down/LOW/0), or X (Don't care).

```
' File [teradyne.vct] created Fri Jun 21 16:58:29 1996
' Teradyne Z18xx vector file for Vector Processor (VP)
' generated by MACHPRO(tm)
' Version 1.35g (c) 1994-1996 Advanced Micro Devices, Inc.

' Pin declaration section
NPINS = 8;
8,Delay_10Ms,I;
7,Delay_01Ms,I;
6,ENABLE      ,I;
5,TRST        ,I;
4,TCK         ,I;
3,TMS         ,I;
2,TDI         ,I;
1,TDO         ,O;
Maxrate 1 MHZ
Mdelay 1000 NS
Thresh LO 1.6 HI 1.6
Term NONE
'===== Begin Vector Section =====
Vector;
Begin Set;
HHHHLLLLX;
HHHHLLLLX;      ' Logic Rst
HHHHLHLX;
HHHHHHHLX;      ' Logic Rst
HHHHLHLX;
HHHHHHHLX;      ' Logic Rst
...
HHLHLLLLX;
HHLHLLLU;      ' Shift DR      'Vector      150000
HHLHLLLLX;
HHLHLLLU;      ' Shift DR
End set;
End Vector;
```

Figure 3. Sample Teradyne Vector File for the Vector Processor

Programming a MACH device requires delays to be inserted at certain points in the vector set. The VP does not have the ability to create these delays so a hardware module has to be added to the test fixture to insert wait states. This module is called the Dual Precise Timer board (Teradyne part number 051-038-00). The DPT drives the VPHOLD line of the VP for either 1 or 10 ms whenever a high to low transition occurs on its A or B inputs, respectively. The MACHPRO-generated vectors contain entries called DELAY_10MS and DELAY_01MS to control the A and B inputs on the DPT.

To program on the Teradyne Z18xx tester, the DPT is wired into the fixture, the vectors added to the local VP library, and digital test steps added to the In-Circuit program. The number of digital test steps is determined by the number of files generated by MACHPRO. An incremental generate operation is performed, and then the digital test steps that program the MACH device must be run in order.

Using the Digital Function Processor

Setting up the DFP to program a MACH is similar to having the DFP program a flash memory. A subdirectory of the board directory is created containing the PT2.INI, PTPROG.EXE and the JEDEC file containing the MACH fuse data. The PT2.INI is edited so it contains the correct device ID, device type, JEDEC filename, translation code, fill data, and chain position. A sample PT2.INI file to program one MACH445 is shown in Figure 4.

```
L,IC1,MACH445,test.jed,91,54096,0,1
R,format 91 = Jedec fuse file
M,0001,07568
R,AMD mfg code=0001(Hex), device
code=07568(Hex)
```

Figure 4. Sample PT2.INI File for PTPROG.EXE Program in DFP

The fields in the PT2.INI file are:

L	= local device tag
IC1	= board identifier
MACH445	= device type
test.jed	= data source file
91	= format of data source file (91 = Jedec fuse File)
54096	= Number of fuses
0	= chain position
1	= fill character
M	= manufacturer tag
0001	= AMD manufacturer code (Hex)
07568	= AMD device code (Hex)
R	= remarks/comments

As long as the fixture is wired according to the comments in the PTPROG.EXE source file no additional modifications are necessary. ProgramVARs is modified to enable the DFP and specify the AUX port and source directory. A DFP worksheet is added and the programming routine is called.

Any time a new JEDEC file is written over the old one, the new JEDEC file will be copied down to the DFP and translated into an image file. This image file will be used by PTPROG.EXE when programming the MACH. Maintaining the test program is easier with a DFP because each time the fuse data/JEDEC file changes, the updates can be automated. If you are using the VP, you can develop a script to call MACHPRO to generate a new set of VP vector files from the new JEDEC file, and then edit the In-Circuit program to add the test steps determined by the number of test vector files created. Check with your local Teradyne Applications Engineer for additional information on using the VP and DFP for programming MACH devices.

Trademarks

Copyright © 1996 Advanced Micro Devices, Inc. All rights reserved.

AMD, the AMD logo, and combinations thereof are trademarks of Advanced Micro Devices, Inc.

MACH and MACHPRO are registered trademarks of Advanced Micro Devices, Inc.

Product names used in this publication are for identification purposes only and may be trademarks of their respective companies.