Contents

Test and Fixture Development

1

Test and Fixture Development

In this chapter...

Objectives

When you finish reading this chapter, you should be able to understand the six steps of the test development process:

- Gather Materials and Define Strategy.
- Create Board Information.
- Generate Tests and Fixture Files.
- Build and Verify Fixture.
- Complete and Debug Tests.
- Release to Production.

This book provides instructions for performing these steps. It is essential that you finish each step completely and correctly before starting the next step.

How to Read this Book

This book contains this introductory chapter, plus one chapter for each step in the test development process. This introductory chapter includes a model of the test development process and provides a brief description of the process steps. Each of the next six chapters describes one step of the test and fixture development process, and provides useful features, including the following:

- a list of the tasks to be performed with instructions for performing those tasks;
- a summary of the step and a list of the new directories and files; and
- detailed information about the software used for each task.

NOTE

Instructions for building the fixture are included in the *Building Board Test Fixtures* documentation.

Detailed information about the tools you use in the test development process is provided in **Test Development Tools**.

The formats of the files involved with developing a board test are described in **Test and Fixture Development**.

You can find complete descriptions of statements in the *Syntax Reference*.

Overview of the Test Development Process

The six steps required to generate a board test are shown under **Objectives** on page 1-1. Follow these steps in the order shown to ensure that the information required for each step is available when it is needed. The more complete and accurate the information is before step three, the less work and re-work there will be in steps three through six.

This documentation describes the test development process sequentially. Your process may involve more than one person and take advantage of parallel effort—that is performing two or more tasks simultaneously.

For example, the **Enter Library Data and Create Custom Library Tests** is parallel task; this task can be accomplished any time after creating the board directory and compiling the `board` file.

The tools we recommend you use to accomplish each task are listed in the explanation of each step.

Brief Description of the Steps

This section provides a brief description of the six steps in the test development process:

1 **Gather materials and define strategy.**

2 **Create board information.**

3 **Generate tests and fixture files.**

4 **Build and verify fixture.**

5 **Complete and debug tests.**

6 **Release to production.**

All steps, including terms and tasks, are fully explained in the corresponding chapters of this user documentation.

1 Gather materials and define strategy.

a Gather together information and equipment, including:

- accurate schematic diagrams and a component locator of the board to be tested;
- material lists identifying the circuit components, pin number diagrams, and component values and tolerances where appropriate;
- at least one known-good PC board (we recommend at least five known-good boards);
- a blank PC board, one not loaded with devices, to be used as an aid in determining pin 1 and 2 of analog devices and in fixture verification (If you do not have an X-Y data file, you will need the blank board for digitizing.);
- adequate pin cards;
- power supplies;
- any needed external instrumentation;
- documentation on all custom and analog and functional devices on the PC board; and
- if you are using Agilent AwareTest xi, a list of devices tested on the AXI system.

b Define the test strategies, such as:

- in-circuit only;
- in-circuit and functional;
- functional, with in-circuit for diagnostics only;
- functional only;
- edge connector only; and
- AXI, in-circuit and functional.

(This is the AwareTest xi strategy.)

NOTE

For detailed information on this strategy, refer to the *Agilent AwareTest xi Process Guide*.

2 Create board information.

a Use BT-BASIC to create `board` and `fixture` directories.

b Use Agilent Board Consultant to create and edit the `board` file. This includes topology of the board and information about the board devices and their interconnections.

c Use Board Consultant to create and edit the `board_xy` file. This includes X-Y locations of device pins, tooling holes, and the outline of the board. You should also include locations of testpads and vias that can be probed.

The `board_xy` file is optional if you are using the Agilent SimPlate Fixture for combinational testing.

d Use BT-BASIC to create the `config` file. This includes a list of testhead resources needed to test the PC board such as pin cards, power supplies, external ports, and external instrumentation.

e Use BT-BASIC to create any necessary custom library tests for devices that have no test in the standard library directories. You can write complete tests or setup-only tests. The setup-only tests include just enough information for the Fixture Generation Software to assign testhead resources. To save time in the test development process, you can complete the setup-only library tests later—while the test fixture is being built, or even after the test and fixture have been released to production.

f When all files are complete and correct, use Board Consultant to compile them. Ensure that the following files are in your board directory before continuing to the next step:

- `board.o`,
- `board_xy.o` (optional for the SimPlate Fixture),
- `config.o`, and
- all necessary custom library test files.

NOTE

AwareTest xi adds new probe reduction techniques to Agilent Access Consultant. If you are using AwareTest xi, you will use these techniques before you build the fixture.

3 Generate tests and fixture files.

a Use IPG Test Consultant to generate the initial fixture files by running the `Board Placement` and `Probe Select` sections of the fixture generation software.

The Board Placement and Probe Select sections of the fixture generation software read the `board.o`, `board_xy.o`, and `config.o` files. They use this information to determine the placement of the board on the fixture and to assign probes.

b Run IPG Test Consultant to generate the tests.

IPG Test Consultant analyzes the board data you provided and generates the device tests, the `testorder` file, the `ipg/summary` file, and the `ipg/details` file.

c Run the Testplan Generator (TPG) to generate the `testplan`.

The TPG uses the `testorder` file to write the test program, which is called `testplan`. It also merges a `testmain` file into the `testplan` to control the testing of the board. TPG looks in the `ipg` directory of the local board directory for a `testmain` file. If it does not find one, TPG uses `$AGILENT3070_ROOT/standard/testmain` (`abhtestmain` for the Agilent Express Cassette fixtures).

NOTE

With Agilent 3070 software revision 3070 04.00pa, an environment variable was created so that files can be easily transferred between UNIX® and MS Windows® controllers, which have different file systems. The environment variable, `$AGILENT3070_ROOT`, replaces the upper path names on both systems. For example, the `$AGILENT3070_ROOT` factory default value is /var/hp3070. In this document, only path names using the environment variable are used. If you must use actual path names, refer to older versions of the documentation. Please see **The Root Directory Environment Variable** in *Administering Agilent 3070 UNIX Systems* for further information.

d Create custom executable tests.

You must create any custom executable tests (either setup-only or complete) before generating the test requirements files. If you have a setup-only test, you can complete it later—while the test fixture is being built, or even after the test and fixture have been released to production.

e Compile the tests to generate the requirements files.

IPG Test Consultant compiles all test files to generate requirements files. The fixture generation software uses the requirements files to assign resources.

f Generate the final fixture files and reports by running the Module Pin Assignment (MPA) and

Fixture Tooling sections of the fixture generation software.

The MPA and Fixture Tooling sections of the Fixture Generation Software read the `board.o`, `board_xy.o`, `config.o`, and the requirements (.r) files. They use this information to assign fixture wiring and to create the fixture files and reports needed to build a test fixture.

g Verify the fixture files and reports to be sure that all information is correct and complete before continuing. When IPG Test Consultant runs the Fixture Generation Software, it also instructs the Plot Generator to produce plot files called `wires.p` and `probes.p` (`wirestop.p` and `probestop.p` for the top plate of the cassette fixtures—Express Cassette and Agilent XG-50 Cassette). You can copy these files to a plotter and use them to check the accuracy of the fixture files and reports.

NOTE

See Chapter 5, **The Plot Generator** in *Test Development Tools* for information about the Plot Generator.

h Examine the `fixture/summary` and `fixture/details` files for errors and warnings.

Make any necessary changes and re-run IPG Test Consultant.

i Use IPG Test Consultant to compile the tests to generate object files.

4 Build and verify fixture.

Use the fixture verification software to verify the wiring of your test fixture. For 20 MHz systems, you should also calibrate your test fixture.

5 Complete and debug tests.

a Verify fixture probe contact with the **CHEK-POINT** feature. Use a known-good board to verify that the board under test is contacting the fixture properly. Edit the `pins` file if necessary.

b Use Analog Debug to debug pre-shorts tests such as potentiometer and switch tests.

c Use a known-good board to verify that shorts and opens testing is accurate, and to edit the shorts file if needed.

d Use AutoAnalog Debug and Analog Debug to evaluate and debug the analog in-circuit tests.

e Turn on the DUT power supplies, step through the setup power supply section of the `testplan`, and adjust the power supply current limit.

- **f** Use Digital Debug to evaluate the digital tests and debug any digital in-circuit or digital functional tests that are not working correctly.
- **g** Use Debug to evaluate and debug the analog functional tests.

6 Release to production.

- **a** Prepare the `testplan` and board directory for production testing.
- **b** Continue to support the test during production testing.
- **c** Use IPG Test Consultant to move the board directory to a location in the Agilent 3070 Series 3 file structure used for production testing.
- **d** Edit the `testplan` to set options, such as datalogging, either on or off according to production requirements.
- **e** Continue to enhance and debug tests as problems are encountered. You can use Statistical Quality Control (SQC) methods to tailor your tests to improve fault coverage or to test throughput. Implement changes to the test and fixture as changes (ECOs) are made to the board under test.
- **f** Remove any unnecessary files and subdirectories from the board directory. We recommend that you copy the contents of your board directory to tape. You can use this tape in the future to re-create your board test if needed. Test Consultant has an Archive Board Directory feature to help you accomplish this.

NOTE

Perform maintenance on your test fixture as explained in the *Building Board Test Fixtures* documentation.

2 Gathering Materials and Defining Strategy

In this chapter...

Objectives

When you finish reading this chapter, you should be able to:

- Prepare the test by gathering the necessary materials.
- Decide how you want to test the devices on the board under test.
- Decide what type of fixture you intend to use to test the DUT.

Prerequisites

Before you begin using this chapter, you should already know:

- The six steps of the test development process.

Required Tools and Materials

To accomplish the tasks in this chapter, you will need:

- Accurate schematic diagrams and a component locator of the board to be tested.
- Material lists identifying the circuit components, pin numbering, and device values and tolerances where appropriate.
- At least one known-good PC board.
- A blank PC board, one not loaded with devices, to be used as an aid in determining pin 1 and 2 of analog devices, and in fixture verification.
- Documentation on all custom and analog functional devices on the PC board.
- A file of X-Y coordinates of the holes in the PC board.
- If you are using Agilent AwareTest xi, a list of devices tested on the AXI system.
- Define the fixture type and size, to determine what fixture default file will be needed.
- Define the probe types, to determine what user-modifiable fixture component files will be required to insure access to all traces on the fixture.

Gather Materials

Prepare the Schematic Diagrams

If you will enter the board data manually, instead of translating CAD files, prepare the schematic diagrams.

1 Assign pin numbers.

a Assign pin numbers (1 and 2) on two-lead components, such as resistors, inductors, and capacitors, that do not normally have any pin distinction.

You can use a blank PC board and an ohmmeter or continuity light to follow traces to help determine pin numbers.

b Write the pin numbers on the schematic diagram.

See **Figure 2-1**. Device pins up or to the left are labeled as pin 1, device pins down or to the right are labeled pin 2.

2 Assign electrical node names.

a Assign a name to each electrical node in the circuit.

b Assign node names to all unused device pins. This is important for shorts and opens testing and future considerations.

c Write the node names on the schematic diagram.

Do not use the following characters in part numbers, node names, and device names. These characters have special meaning in the shell or in the `board` file. These are:

* . " ' ` , : [ ] < > | & $ \ !

/ blank ; ? ^ ( )

# - (in first character position)

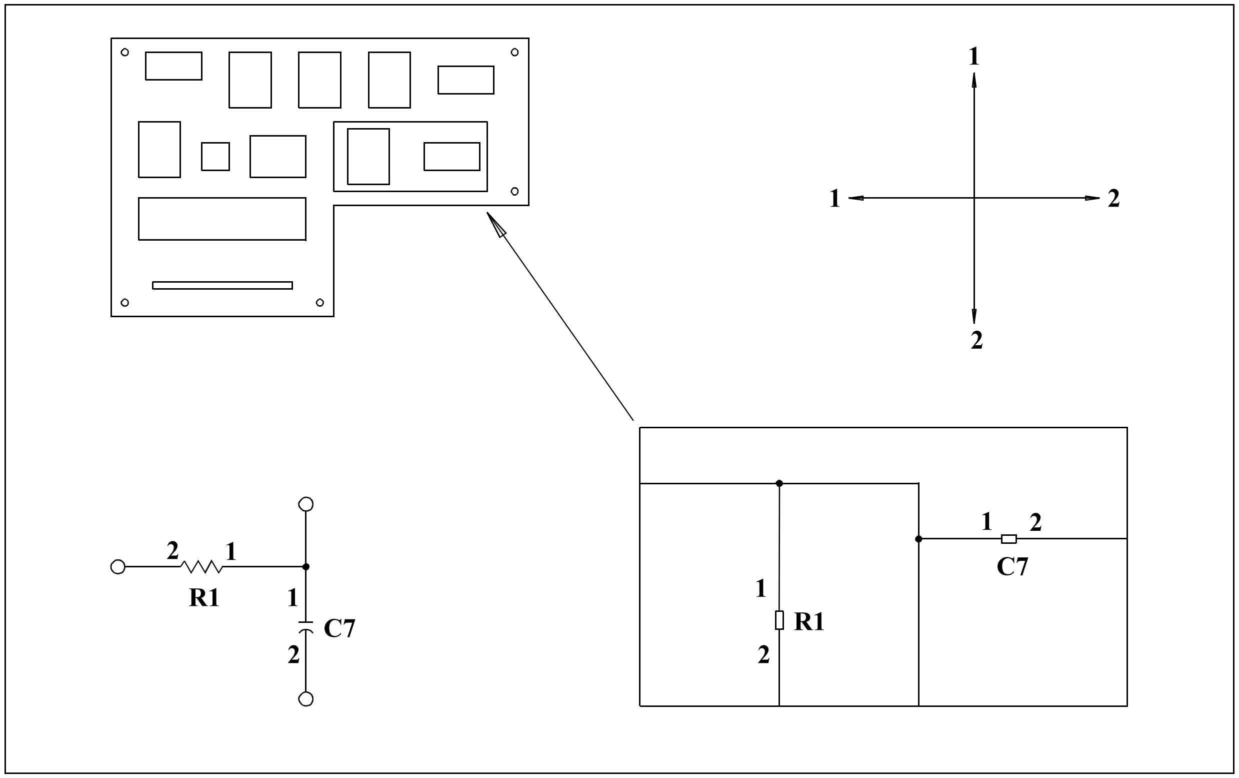
The underscore (_) works well in place of a period (.). Do not use the tilde (~) in device names, node names, or failure messages.

The following node naming conventions are recommended:

- If the node includes one digital device, use the digital device pin in the node name; for example, U110-12.
- If the node includes more than one digital device, use the device pin of the driving device; for example, U2-4.
- Use the signal name for each line of data and address buses, **DATA1**, **ADD4**.
- Use the voltage level of power nodes; for example, +24V.

- Avoid using only numbers for node names because using them would make the reports and files confusing.

Figure 2-1 Device pin numbering



Define Test Strategy

Decide what test strategy to use on your PC board. This involves more than just deciding how to test individual components and circuits. The types of tests you choose to perform depend on whether you want more throughput or a more thorough test. Do you have a hot mock-up? Are you testing at speed? The answer to these and other questions determines what test strategy you employ.

Possible test strategies include:

- in-circuit only;
- in-circuit and functional;
- functional, with in-circuit for diagnostics only;
- functional only;
- edge connector only; and
- AXI, in-circuit and functional. (This is the AwareTest xi strategy.)

Consider the optional test methods listed in **Table 2-1**.

Table 2-1 Optional test methods

Method	Description
Agilent PanelTest	An fast and easy method to develop tests and fixtures for multiple board panels.
Agilent Throughput Multiplier	A method to test boards simultaneously (one board per testhead module) to increase test system throughput.
Dual-Well Shared Wiring	A method to reduce the number of testhead resources required, and increase operator throughput by using shared (parallel) wiring from resources to boards under test.
Agilent TestJet	An unpowered test of the connectivity of device pins.
Agilent Polarity Check	An unpowered test of the orientation of electrolytic capacitors.

Table 2-1 Optional test methods (continued)

Method	Description
Agilent Connect Check	An unpowered test of the connectivity of device pins.
Agilent Multiple Board Versions	A method to use one test and fixture to test multiple versions of a PC board.

Table 2-2 presents items to consider when you are determining the test strategy. This is not an exhaustive list; rather, it is intended to show you the kinds of things you should consider.

Table 2-2 Determining test strategy

Consideration	Issues
Board volume	If many boards of a given type are tested, the per-board cost of developing the test is relatively small. This makes it practical to develop a comprehensive test that includes both in-circuit and functional testing. When only a few boards of a given type are tested, the test cost per board is high, so an in-circuit only test might be preferable. If test time is a concern, high volume applications may need a faster test than low volume applications. A faster, in-circuit test usually means less fault coverage and accuracy than a slower, more thorough test. One advantage of having a large volume of a particular board is that you can quickly collect meaningful fault data and apply SQC methods to improve your production process.

Table 2-2 Determining test strategy (continued)

Consideration	Issues
Fault spectrum	The kinds of faults on the PC boards depend on such things as the types of components and how they are used, the manufacturing process, and how the board was designed. Process faults and digital device problems are best found by in-circuit testing. Timing faults in high speed circuits require functional testing.
Yield into test system	This is the percentage of your boards which you expect to be good when they arrive at the test system. If your yield into the test system is low, you will want a thorough in-circuit test because of its ability to diagnose faults easily. If your yield is high, you can rely on less in-circuit testing. In either case, you should enable datalogging to collect data that can be used to produce SQC reports.
Programming capacity	The size of your programming department and its current workload should be considered. In-circuit tests require less programming time than functional tests and are easier to generate and debug. If desired, you can develop only an in-circuit test and start testing boards as soon as possible, then develop the functional part of the test later. To avoid modifications to the fixture, provide at least setup-only tests so that resources will be assigned and built into the fixture for any future testing.
Quality objectives & company test strategy	These two considerations are closely tied together, with the strategy generally based on such quality objectives as the required test system yield. Different companies will have different objectives and strategies. The environment in which the product is to be used can also affect the test strategy. If the boards are to be used in a critical environment, exhaustive in-circuit and functional testing may be justified.

To define the test strategy:

1 **Learn the function of the board and its components.**

2 **Identify the clusters to be functionally tested.**

(Digital testing only) It is better to have several small clusters than a few large ones. Smaller circuits let the fixture generation software make more efficient use of testhead resources and require fewer module cards to implement the test. Also, diagnosing faults within small clusters is easier than diagnosing faults within large clusters.

3 **Determine device dependencies.**

Arrange the testing according to device dependencies. For example, if device A will not work unless device B is working, device B needs to be tested first.

4 **Determine digital device intrinsics.**

Determine digital device intrinsics, for example, determine the internal resistance between two pins of the device. Model the device in the board description with an internal device entry or a Part Description Library.

5 **Determine which fault types to cover.**

Decide which fault types your test will identify. For example, do you need to diagnose which RAM cell is faulty or is it enough to know that the RAM chip is not working? If your board has a device that can operate at a higher speed than the circuit requires, is it necessary to test that device to its high-speed specification?

6 **Determine Fixture Strategy.**

- Possible fixture strategies are:
  - Top side probing,
  - Wireless fixtures,
  - Short wire fixtures,
  - Long wire fixtures,
  - Fixture electronics,
  - Throughput multiplier,
  - Dual well shared wire,
  - Dual stage fixturing,
  - Vacuum fixtures,
  - Neumatic fixtures,
- Which fixture default files are needed
- If user-modifiable fixture component files are needed.

7 **Document the strategy.**

3 Creating Board Information

In this chapter...

Objectives

When you finish reading this chapter, you should be able to:

- Create the directory for the board test directories and files. This is the Local Board Directory.
- Create the board description file `board` and the X-Y location information file `board_xy`.
- Create the board configuration file `config`.
- Enter library data and create any library tests for devices that have no test in the standard library directories.
- Compile the board (`board`) and X-Y data (`board_xy`) files.

Prerequisites

Before you begin using this chapter, you should already know:

- how you want to test the devices on the board under test.

Required Tools and Materials

To accomplish the tasks in this chapter, you need:

- an Agilent 3070 Series 3 work station.

Related Documents

- *Agilent No-Wire Technology Test and Fixture Development* manual.

Create the Board Directory

Create a directory for the board test files. This directory is called the `board` directory. You can name the directory structure to fit your specific process or system. The examples shown in this chapter are general.

NOTE

Note that this directory structure for a multiple-board fixture is NOT the same as for Agilent PanelTest. If you are developing a test and fixture for PanelTest, see Chapter 1, **Multiple-Board Tests & Fixtures** in the *Optional Board Test Applications* documentation.

Board and Fixture Directory Structure

Although you can create a local board directory (and its related files) anywhere in the file structure, we recommend you do the following:

- While developing a board test, create the local board directory beneath the home directory associated with your login. This is to avoid using disk space on the production system and to be sure that only final files and tests exist on the production system. For example, if you use the **user1** login, your local board directory might be called `/users/user1/my_board`.

NOTE

Choose the board name, `my_board`, carefully because it becomes the board ID and remains the same when released to production.

- When the board test is finished and ready to be released to production, move your local board directory to the `$AGILENT3070_ROOT/boards` directory. For example, the directory might then be called `$AGILENT3070_ROOT/boards/my_board`.

NOTE

With Agilent 3070 software revision 3070 04.00pa, an environment variable was created so that files can be easily transferred between UNIX® and MS Windows® controllers, which have different file systems. The environment variable, `$AGILENT3070_ROOT`, replaces the upper path names on both systems. For example, the `$AGILENT3070_ROOT` factory default value is /var/hp3070. In this document, only path names using the environment variable are used. If you must use actual path names, refer to older versions of the documentation. Please see **The Root Directory Environment Variable** in *Administering Agilent 3070 UNIX Systems* for further information.

Refer to **Figure 3-1** on page 3-5 and consider the cases of two different fixtures:

- a single-board fixture to test one board (`board_1`), and
- a multiple-board fixture to test two boards (`board_2`) and (`board_3`).

There are two sets of files to be concerned with:

- board files, which are board description files and test files; and
- fixture files, which are fixture building files and reports.

The files for the single board are stored under the `board_1` directory. The files for the multiple board are stored under the `multbrd` directory. Multiple boards are linked to one fixture directory. Each directory has its own fixture directory for fixture reports and files.

The `board_1` directory has one set of board files and one set of fixture files. The `multbrd` directory needs two sets of board files (one for each board) and only one set of fixture files. To simplify the diagram, not all board files are included in **Figure 3-1** on page 3-5.

To keep the file structure simple and to avoid duplicate files, the system uses symbolic links for the `multbrd` directory. You can find, use, and edit the fixture and board files under each board directory, `board_2` and `board_3`. There is, however, only one set of fixture files stored under `boards/multbrd/fixture`. This is accomplished by the symbolic links.

The board and fixture files for `board_1` are stored under the directories:

```
$AGILENT3070_ROOT/boards/board_1
$AGILENT3070_ROOT/boards/board_1/fixture
```

The board files for `multbrd` are stored under the directories:

```
$AGILENT3070_ROOT/boards/multbrd/board_2
$AGILENT3070_ROOT/boards/multbrd/board_3
```

Links to these directories from the corresponding board directories are found at:

```
$AGILENT3070_ROOT/boards/multbrd/fixture
/boards
```

The fixture files for `multbrd` are stored under the directory:

```
$AGILENT3070_ROOT/boards/multbrd/fixture
```

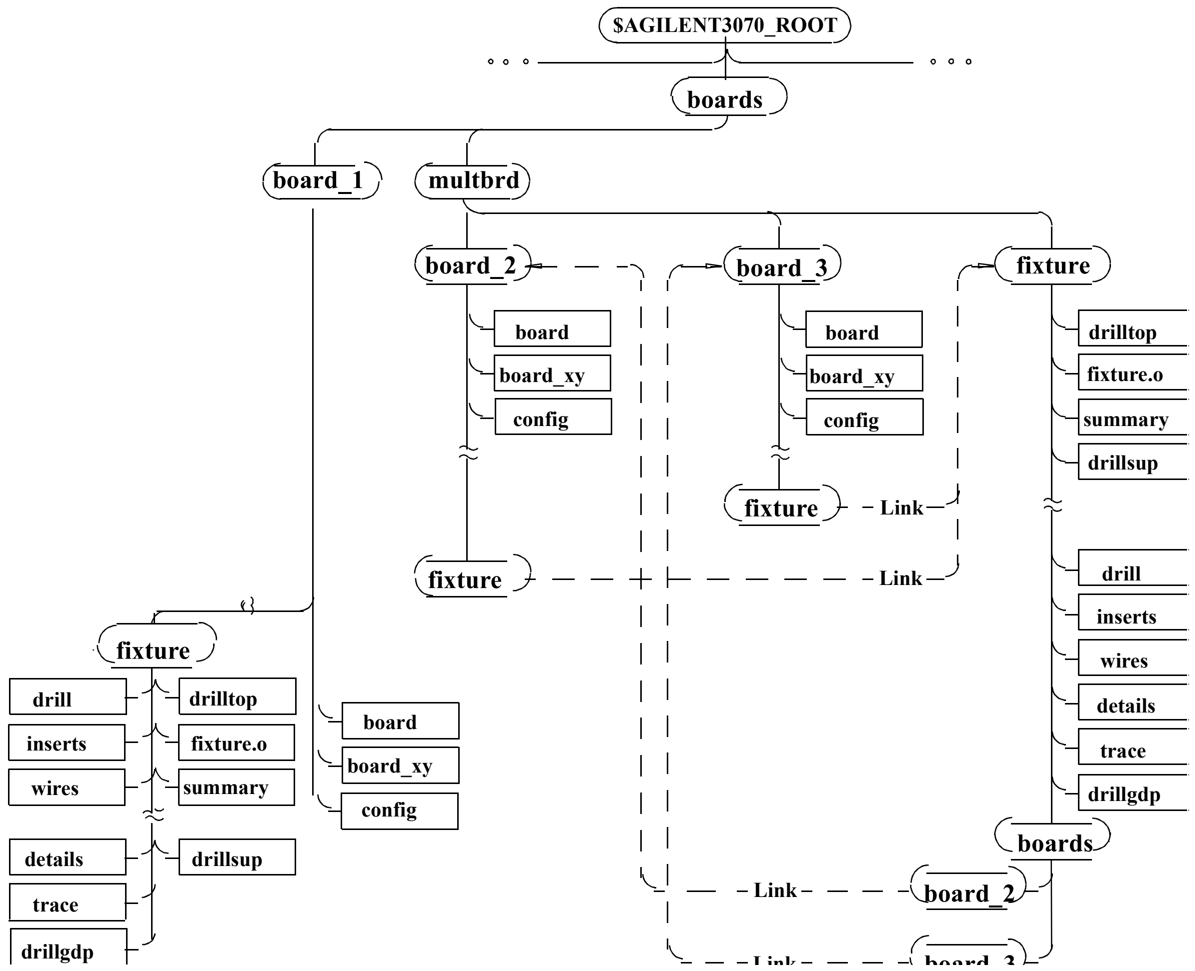
There are links to the fixture directories under `board_2` and `board_3` from this directory.

If you need to use custom fixture `component` files that are applicable to all boards on a multiboard fixture, then you should install the `custom_fix sub`directory in the `fixture` directory. If the component files are only applicable to one of the boards then you should place the `custom_fix` subdirectory in the applicable board directory. You may also have different `custom_fix` sub directories for each board on the fixture.

NOTE

If you are not familiar with the structure of the file system, see Chapter 3, **The File System** in *Board Test Fundamentals*.

Figure 3-1 Board and fixture directory structure



Log On to the 3070 Series 3 System

When you log on to the 3070 Series 3 system as a test developer (user1, for example) you are automatically in HP CDE or the console environment, depending on which system controller you are using.

There are two ways to create the local board directory.

- If you are using a single-board fixture, see **Create the Board Directory for a Single-Board Fixture** on page 3-6.
- If you are using a multiple-board fixture, see **Create the Board Directory for a Multiple-Board Fixture** on page 3-6.

Create the Board Directory for a Single-Board Fixture

1 **Create the local board directory.**

   a In BT-BASIC, type:

   ```
   create dir
   ```

2 **Go to the new directory.**

   Use the `msi` command in BT-BASIC to change to a different directory.

   a For example, if your directory is `/users/user1/my_board`, then type:

   ```
   msi btgetenv$("AGILENT3070_ROOT") &
     "/users/user1/my_board"
   ```

You have now created the local board directory and are switched to that directory.

Create the Board Directory for a Multiple-Board Fixture

NOTE

For multiple boards use PanelTest. See “Agilent PanelTest” in the *Optional Board Test Applications Manual*.

Create Fixture Defaults and User-modifiable Fixture Component Files

Both the Fixture defaults and user-modifiable fixture component files give the test developer greater control over the fixture initialization and node access.

CAUTION

⚠ You should consult with your fixture vendor before making any changes to `fixture component` files. Failure to do so could result in drilled probe plates that cannot be populated with probes, or may add significant costs to building the fixture.

CAUTION

⚠ Both the optional Fixture Defaults file and user-modifiable fixture component files must be defined before Board Consultant or IPG Test Consultant generates the fixture files.

If changes are made to the Fixture Defaults file or fixture component files after fixture files are created, you MUST remove the fixture files and restart test development.

Fixture Defaults File

The fixture default file syntax and use model is described in Chapter 5, **Fixture Defaults File** in the *Data Formats* documentation.

User-Modifiable Fixture and Probes Files

The user-modifiable fixture and probes files syntax and use model are described in Chapter 5, **User-Modifiable Fixture Components** in the *Data Formats* documentation.

If user-modifiable fixture components meet your fixturing needs, do the following:

1. Determine if a single fixture, or many of your fixtures will use this functionality. Set up the appropriate directory structure. See **Define the custom_fix Directory** on page 3-8.
2. Copy one or more standard fixture component files to the directory.

NOTE

Do NOT change the name of the component file(s).

3 Change whatever attributes are desired for board development. For the probe files:

- The usage field gives the developer better control over which probes are used. For example, the 75 mil probe default value is PREFERRED, but the user can override it by setting the usage to MINIMIZE. This will cause the probe select software to only use 75 mil probes where it is absolutely required, instead of almost everywhere.
- At least one LONG probe and one regular probe must have a usage of `available`. If this is not the case then the software will error.
- If the usage enumeration is in conflict with the `.hp3070` options to use 75mil and 50mil probes, then an error will be generated. To avoid this, comment the options for 50 and 75 mil probes in the `.hp3070` file.
- FXT will use attributes for a probe that has been marked unavailable. If an unavailable probe is found in the fixture object, then it assumes the user manually added it, and FXT will utilize the probe's attributes for generating the construction files.

4 To enable the use of dense probe spacing, add the following enable statement to the board config file:

```
enable advanced fixturing
```

When this statement is used, the software will calculate probe spacing using an aggressive spacing algorithm. See **Understanding Probe Spacing** on page 3-11. If this statement is not present, the Agilent3070 fixturing software will use conservative spacing as it did prior to the 05.00 version.

Define the `custom_fix` Directory

Normally you will create a directory named `custom_fix` in the local board directory. This is similar to the way `custom_lib` is used for custom libraries, except that fixture component files are placed in the `custom_fix` directory.

If you will have a group of fixtures, you may want to create a special `custom_fix` subdirectory in a location shared by all boards. For example, `$CUSTOM_FIX=$AGILENT3070_ROOT/boards/custom_fix`.

The `custom_fix` directory can also reside in the local board fixture directory or the fixture directory used for "multiple board per fixture" (mbpf) applications. The search precedence for finding fixture component files is:

1 Directory defined in environmental variable `$CUSTOM_FIX`.

2 <board dir>/fixture/custom_fix or <mbpf fixture dir>/custom_fix

3 <board dir>/custom_fix

NOTE

For mbpf fixtures, you cannot have the `custom_fix` directory in the board directories.

4 $AGILENT3070_ROOT/standard/fixture/components

How the Fixture Component Files are Utilized

The software will read fixture build parameters from Agilent-supplied and user-modified component files instead of using hard-coded software constants.

When using fixture component files, the software will search directories looking for fixture component files in the order listed above. The first file found for a specific type will take precedence over files found later on. All files must be accounted for; if one is missing, an error will be reported. This should never occur as all the files should be found in `$AGILENT3070_ROOT/standard/fixture /components`.

Once a file is found, all fields must be present and they must have assigned values. Syntax and semantic checking is performed to validate the attributes. Any errors found will be reported in the fixture/details file.

Any files found in the first three directories in the search order are treated as modified components. The contents of these files are echoed to the fixture/details file as a flag to the user that custom attributes are being used. The directory path of each custom fixture component file is also written to the details file.

Validating Fixture Component Changes

CAUTION

⚠ If a fixture directory already exists, back it up by moving it to a new name or location. You MUST remove the fixture.o file to have the generate process show the result of adding fixture defaults and user-modifiable probes. The fixture directory will have some files overwritten during the following process.

The intent of this discussion is to outline a procedure for validating any changes made to fixture component files before doing full fixture development.

Changing the `Diameter` or the `CollisionDiameterMils` of a probe will impact minimum probe to probe clearances.   These changes should be done with caution because it may increase the cost of building and maintaining your fixture.

To understand the ramifications of any changes made to the component files it is recommended that you execute the following procedure:

1 **Compile the config, board, board_xy, fixture defaults, and any custom libraries.**

ADVICE

Failure to compile the fixture defaults file will cause the software to ignore the optional fixture defaults files. This was done for backward compatibility.

You may manually compile the files or use **Agilent IPG Test Consultant** to compile them.

2 **Manually run Board Placement.**

See **How to Run Board Placement** on page 4-66 for more information.

3 **Manually Run Probe Select**

See **How to Run Probe Select** on page 4-68 for more information.

4 **Open the fixture/details File.**

Examine the clearances outlined in "Calculated Probe to Probe Clearances" table. A sample report for the standard fixture component files shipped with the system is shown in **Example 3-1** on page 3-11.

Please note that Probe Select will generate the table even if errors occur.

The sort order of the report is significant. It is based on the usage field in the component files and clearances computed. The sort order shows what component pairs will be used first when a match is found. For example, two probe locations spaced 64.0 mils apart may be probed using a 100 mil and a 50 mil probe, as indicated by the bold text in **Example 3-1**. Also see **Understanding Probe Spacing** on page 3-11.

5 **Consult with your fixture vendor and make sure they are able to build the fixture with the modified clearances.**

6 **Steps (2) to (5) may have to be repeated several times before achieving usable results.**

For each iteration, the fixture directory should be removed before validating new changes to the fixture component files.

7 **If you choose not to use the new fixture files, with the fixture defaults and user-modified probes, restore the backup fixture directory to the original board directory.**

Example 3-1 Probe to Probe Minimum Clearance Report

```
          CALCULATED PROBE TO PROBE CLEARANCES
          ------------------------------------

    P r o b e    t o    P r o b e       Min Clearance
--------------------------------------  -------------

 Long     probe - Long     probe         1000
 100 Mil probe - Long     probe          915
 100 Mil probe - 100 Mil probe           850
 75  Mil probe - Long     probe          835
 75  Mil probe - 100 Mil probe           770
 75  Mil probe - 75  Mil probe           700
 50  Mil probe - Long     probe          705
 50  Mil probe - 100 Mil probe           640
 50  Mil probe - 75  Mil probe           570
 50  Mil probe - 50  Mil probe           490
 Long     probe - 375 Mil tooling pin    2895
 Long     probe - 200 Mil tooling pin    1495
 100 Mil probe - 375 Mil tooling pin     2830
 100 Mil probe - 200 Mil tooling pin     1430
 75  Mil probe - 375 Mil tooling pin     2760
 75  Mil probe - 200 Mil tooling pin     1360
 50  Mil probe - 375 Mil tooling pin     2680
 50  Mil probe - 200 Mil tooling pin     1280
```

Understanding Probe Spacing

There are two ways for determining the probe to probe spacing. If the board config file specifies `enable advanced fixturing`, the aggressive advanced probe spacing will be used. If the board config file does not specify `enable advanced fixturing`, the conservative standard spacing will be used.

The minimum spacing is determined by:

- Fixture plate tensile strength, i.e. how much material is required between probes during insertion to prevent plate damage.

- Insertion tool required diameter

The Probe Select program uses the selected algorithm for each probe to determine minimum spacing in an interactive loop. The spacing is modified by choosing specific probes attributes, as described in *Data Formats* documentation, **Fixture Components Files Syntax** on page 5-60.

Advanced (Aggressive) Probe Spacing

NOTE

The order the probes are placed is a major factor in determining the spacing. When a large probe is inserted first, the tool can overlap where the smaller pin will be inserted. The reverse order of insertion will not allow the tool to push the probe in, because it will bend the smaller probe.

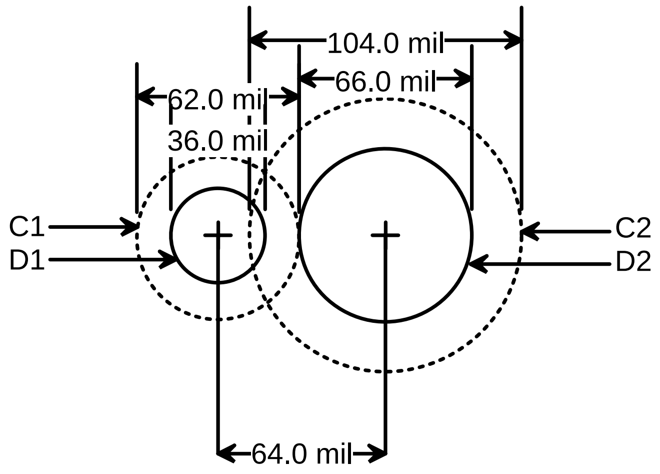
The algorithm for determination of minimum probe spacing is:

```
    S1 = (C1+D2)/2
    S2 = (C2+D1)/2
If S1<S2
    then spacing = S1
    else spacing = S2
```

For example, as shown in **Figure 3-2**, minimum probe spacing is:

```
S1 = (62+66)/2 = 64 mils
S2 = (104+36)/2 = 70 mils
    S1<S2
    Spacing = S1 = 64 mils
```

Figure 3-2 Advanced Probe Clearance 100 Mil probe to 50 mil probe



Standard (Conservative) Probe Spacing

The algorithm for determination of minimum probe spacing is:

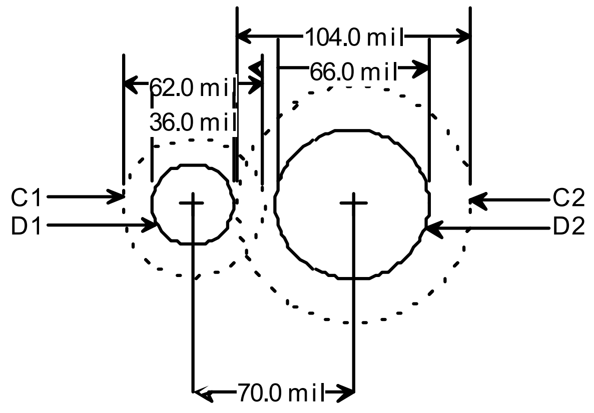
```
    S1 = (C1+D2)/2
    S2 = (C2+D1)/2
If S1<S2
    then spacing = S2
    else spacing = S1
```

For example, as shown in **Figure 3-2**, minimum probe spacing is:

```
S1 = (62+66)/2 = 64 mils
S2 = (104+36)/2 = 70 mils
   S1<S2
   Spacing = S2 = 70 mils
```

In some cases 1-3 mils are added to the calculated spacing to be compatible with pre 05.00 software.

Figure 3-3 Standard Probe Clearance 100 Mil probe to 50 mil probe



Create the Board and X-Y Data Files

Provide the system with information about the board to be tested. The information includes:

- what devices are on the board;
- how the devices are connected;
- the X-Y locations of the devices;
- the board outline; and
- the tooling pin hole locations.

This information is included in two files: the board file `board`, and the X-Y data file `board_xy`.

NOTE

For an explanation of the format and syntax of these files, see Chapter 1, **The Board File** and Chapter 2, **The board_xy File** in *Data Formats*.

Use Agilent Board Consultant to create and edit the `board` and `board_xy` files. You can also create a `board` file by translating an Agilent 3065 BCF file with the `translate board` statement.

Edge Connector Testing

An edge-connector-only test is simply a cluster test of the entire board. This tests the functionality of the circuit board by probing only the edge connectors or a limited number of strategic nodes of the circuit. This is useful when many of the nodes of the circuit are not accessible by the test fixture, or when the probes and wiring of the fixture would load the circuit and change its operation or characteristics.

To develop an edge connector test:

1. **Set the board defaults.**
2. **Enter the board information.**
3. **Specify the test strategy.**
4. **Create the test.**

1 Set the board defaults.

a Use one of the following methods to set the board defaults:

- Before invoking Board Consultant, copy the edge-connector-only defaults file to your local board directory. For example:

```
cp
$AGILENT3070_ROOT/standard/edge_board
_defaults /var/board_defaults/
```

  Board Consultant uses the `board_defaults` file in your local board directory.

- Specify the `edge_board_defaults` file as the board defaults file when you run Board Consultant.

NOTE

See Chapter 7, **The Board Defaults Files** in *Data Formats* for a description of the `board_defaults` files.

2 Enter the board information.

Enter the board information. If all devices on the board are to be considered internal devices, you do not need to enter them with the **Internal Devices Entry** form.

3 Specify the test strategy.

a In IPG Global Options, select **Edge Connector Only**.

The test strategy instructs the fixture generation software to produce appropriate files and reports to build the fixture for edge-connector-only testing.

If you want to develop an edge-connector-only test which, when it fails, performs some in-circuit testing on the board, specify the test strategy as **Combinational**. You can develop a combinational test and include a cluster for the entire board under test. You can then arrange the testplan to perform the testing desired.

4 Create the test.

Test the entire board as a cluster.

NOTE

To develop an edge-connector-only test, you must supply a `board_xy` file even if you are using the Agilent SimPlate fixture. You can use asterisks (**) in place of X-Y coordinates in the `board_xy` file for the SimPlate fixture.

NOTE

Always perform shorts and opens testing before applying power to the board under test.

Run Board Consultant

1 Start Board Consultant.

a From BT-BASIC, type:

```
board consultant
```

or,

b From the IPG Test Consultant main menu, select **Enter Board Data.**

2 Remove the BT-BASIC window or IPG Test Consultant from the screen.

a Use one of the following methods to clear the BT-BASIC window or IPG Test Consultant from the screen.

- Exit from the BT-BASIC window or IPG Test Consultant.

  If you choose to exit from BT-BASIC and IPG Test Consultant, you can run Board Consultant from HP CDE or from the console environment.

  or,

- Minimize the applications after starting Board Consultant.

NOTE

This chapter refers to the flow chart elements as blocks; it refers to the choices in the list below the flow chart as actions.

NOTE

If you do not have the optional PanelTest software installed on your system, the Board Consultant flow chart does include the two blocks called: **View / Edit Panel Data** or **Board Type**.

Create the Data Files

1. **Create new board.**
2. **Translate CAD Data.**
3. **Load existing board.**

1 Create new board.

a Select the **Create New Board** block

Create board data files from start without translating other data files.

b Specify the board directory.

c Select the **Create Board** button to create the files and return to the flow chart.

2 Translate CAD Data.

Agilent recommends that you use CAMCAD to translate your board files. Find information on this product on the internet at www.camcad.com.

3 Load existing board.

a Examine the displayed graphics to check that:

- CAMCAD translated the data files for the correct board,
- the tooling pin holes are correct,
- the board outline is correct,
- the board keepout areas are free of probes, and
- the devices and probe locations are on the correct side of the board (top or bottom).

b Select the **Load Existing Board** block to resume creating board data files after interrupting the process or after running CAMCAD.

c If needed, use the displayed Board Specification Form to specify the board data files.

d Select the **Load Board** button to load the board files and return to the flow chart.

e If you want to temporarily stop the test development process:

- Select **File** from the top of the flow chart window.
- Select **Save Board Information** to save the data that you have entered.
- Exit Board Consultant.
- Start Board Consultant to resume the process and start with the **Load Existing Board** block.

View / Edit Physical Board Data

1 Select the View / Edit Physical Board Data block.

Begin manually creating the `board` and `board_xy` files, or edit existing `board` and `board_xy` files.

2 Examine the displayed graphics.

- Board outline
- Tooling pin holes
- Board keepout areas

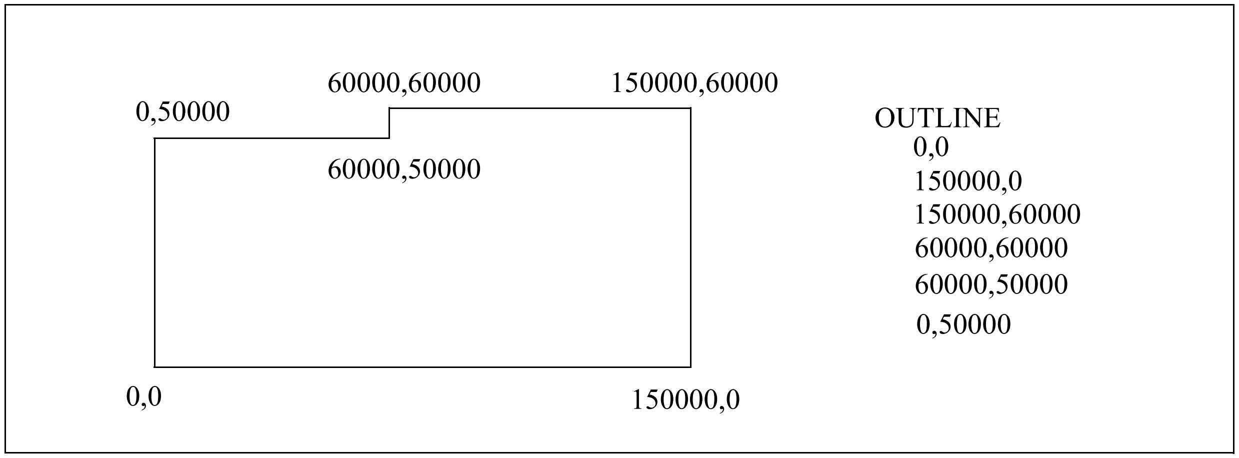
a Select the **Enter Board Outline** action.

b Use the displayed form to enter or edit the outline coordinates of the board to be tested; ignore any notches or holes in the board outline.

The board outline is specified by the series of points that define the corners of the board. As shown in **Figure 3-4**, the bottom left corner of the board is considered the board origin with X-Y coordinates of 0,0. The other points are listed as X-Y coordinates relative to the board origin.

The points are listed starting at the board origin and continuing around the board in a clockwise or counter-clockwise direction. The board outline may or may not include the origin again at the end of the list.

Figure 3-4 Board outline



c Select **Enter Tooling Holes**.

Use the displayed form to enter or edit the coordinates and diameter of the board's tooling pin holes. The tooling holes are specified as X-Y coordinates relative to the board origin. The diameter can be 2000 or 3750. Enter 2000 to specify the fixed diameter pin; enter 3750 to specify the variable diameter tooling pin.

View / Edit Board Description

1 **Select the View / Edit Board Description block.**

2 **Select devices.**

3 **Enter or edit the device data.**

1 Select the View / Edit Board Description block.

The data that you enter or edit under this selection is shown in **Table 3-1**.

Table 3-1 Data entered in the View / Edit tooling holes form

Data	Description
Devices	Devices on the board and their connections, values, X-Y locations, probing attributes, and device testing options.
Internal Devices	Devices that are elements of a package part or of a cluster.
Nodes	Electrical nodes on the board and their connections.

2 Select devices.

Select the appropriate action for the device that you want to enter or edit. The device types are:

- capacitor,
- connector (for shorts and opens testing only--there is no test for connectors),
- diode,
- FET (depletion mode junction FETs only),
- fuse,
- inductor,
- jumper/strap,
- node library (devices for which you write node-oriented tests),
- pin library (devices for which you write pin-oriented tests or part description library tests),
- potentiometer,
- resistor,
- switch,
- transistor,
- zener,
- node, and
- internal devices.

3 Enter or edit the device data.

The device entry form includes entry fields for values and options applicable only to the device type selected. (You may enter up to 8,000 devices.)

NOTE

We recommend that you enter all pins on each device, including power and ground pins; this is mandatory for devices to be tested with Agilent TestJet.

a Enter the Device Designator, Part Number (if applicable), and Side of Board.

NOTE

Do not use the following characters in part numbers, node names, and device names.

* . " ' ` , : [ ] < > | & $ \ ! / blank ; ? ^ (
) # - (in first character position)

Do not use the tilde (~) in device names, node names, or failure messages. These characters have special meaning in the shell or in the `board` file.

b Enter the Device Values and Tolerances.

c Enter the Connections.

d Enter the Test Options.

The values and options displayed on the entry form are:

- Translated by CAMCAD from data files.
- Included in the `board_defaults` file.
- Specified in IPG Global Options.

The device options are shown in **Table 3-2** on page 3-21.

Table 3-2 Device-specific IPG global options

Option	Description
Tolerance Multiplier	Specifies how accurately a device should be tested. This can be from 0.1 to 10. The default is 5. IPG uses this number to determine how accurate a test to write. The smaller the value of the tolerance multiplier, the more accurate the test. More accurate tests require more test time and resources.
Remote Sensing	Indicates if remote sensing should be used. This can be **ON** or **OFF**. The default is **ON**. IPG uses this value to determine if it should use remote sensing if needed, or not at all. Remote sensing results in more accurate tests, however, it requires more test time and resources.
Adjust	Specifies the type of adjustment to be used when testing adjustable devices. This can be **NONE**, **ACCURATE**, or **FAST**. The default is **ACCURATE**. IPG uses this value to generate tests of adjustable devices. If you specify **NONE**, the device is tested without allowing the operator to adjust the part. If you specify **ACCURATE**, the operator is prompted to adjust the device every time it is tested. If you specify **FAST**, the operator is prompted to adjust the device only if it does not measure within tolerance.
Diode Current	Specifies the maximum current (in amps) to be used in a diode test. The default is 5mA. IPG uses this value to generate diode tests.
Fuse Threshold	Specifies the resistance value (in ohms) to be used to determine the presence of a fuse. The default is 10 ohms. If the resistance of the fuse is less than or equal to the threshold, the fuse is considered present. IPG uses this value to generate fuse tests.
Upstream Disable	Specifies if IPG should try to disable upstream devices. This can be **OFF** or **ON**. The default is **OFF**: do not disable.

Table 3-2 Device-specific IPG global options (continued)

Option	Description
Upstream Condition	Specifies if IPG should try to condition upstream devices. This can be **OFF** or **ON**. The default is **OFF**: do not try to condition upstream devices. If both **Upstream Condition** and **Upstream Disable** are turned on, IPG tries to disable the upstream device; if the device cannot be disabled, IPG tries to condition the device.
Zener Current	Specifies the maximum current (in amps) to be used in a zener test. The default is 10mA.
Testable	Indicates if the device is to be tested. This can be **YES** or **NO**. The default is **YES**: test the device.
Safeguard	Indicates if SAFEGUARD is to be considered for this device when testing other devices to which it is connected. This can be **YES** or **NO**. The default is **YES**, consider the SAFEGUARD.
Backtrace	Indicates if backtracing is to be used in the test. This applies only to digital functional tests. This can be **YES** or **NO**. The default is **YES**: use backtracing.
Testability Standard	Specifies that the device is a boundary-scan device.
BSDL Part Number	Specifies the part number of the boundary-scan device.
Device Package Type	Specifies the package type of the boundary-scan device.
Library Test Expected	Indicates if the device is to be tested with a digital test in addition to the boundary-scan or Agilent TestJet test. This can be **YES** or **NO**. The default is **NO**: do not perform the digital test.
Connect Test	Specifies that the device is to be tested with a boundary-scan connect test. This can be **YES** or **NO**. The default is **YES**: perform a boundary-scan connect test.

Table 3-2 Device-specific IPG global options (continued)

Option	Description
Interconnect Tests	Specifies if the device is to be tested with a full boundary-scan interconnect test, or if the device is only to be included in a chain. This can be **FULL** or **TAP ONLY**. The default is **FULL**.
Test Using Agilent TestJet	Specifies whether the device is to be tested with an Agilent TestJet test. This can be **YES** or **NO**.

e Specify the Probe Location Attributes.

The probe select stage of the fixture generation software chooses the best probe location for each node, electrically and physically, from the list of possible locations for each node.

Board Consultant places the attributes in the `board_xy` file; the fixture generation software reads the attributes from the `board_xy` file and places them in the `fixture.o` file.

The probe location attributes are grouped into three categories as shown in **Table 3-3**.

- Enter the Probe Access Attributes. These are specified in **Table 3-4** on page 3-24.
- Enter the Fixture Access Attributes. These are specified in **Table 3-5** on page 3-26.

Enter the Manual Access Attributes. These are specified in **Table 3-6** on page 3-26.

Table 3-3 Probe location attributes

Attribute	Description
Probe Access	Attributes that affect the location of fixture probes.
Fixture Access	Attributes that specify the accessibility of the probe locations by the fixture probes.
Manual Access	Attributes that specify the accessibility of the probe locations by the guided probe.

Table 3-4 Probe access attributes

Attribute	Description
MANDATORY	Mandates the use of a specific location. You can specify multiple **MANDATORY** locations on each node. You can specify the use of a top-side location by labeling it as **MANDATORY**. If Probe Select cannot use the mandatory location, it uses one of the alternates and issue a warning. Locations that are also specified in a GROUP must be specified as **MANDATORY**.
PREFERRED	Specifies that this location should be used unless it causes a mechanical conflict such as a conflict in density or accessibility. **PREFERRED** overrides all electrical considerations. You can specify more than one location as **PREFERRED**; Probe Select chooses from preferred locations first. If a preferred location cannot be used, Probe Select uses an alternate location.
NORMAL	Instructs Probe Select to evaluate this location using its own algorithm. If no probe access attribute is specified, **NORMAL** is assumed.
UNRELIABLE	Instructs Probe Select to use this location only if there is no other possibility because it is a physically unreliable place to probe.
EXTRA	Specifies that the location should not be used as a probing point but should be passed on to the `fixture.o` file as an alternate location for future consideration.
NO_PROBE	Indicates a location that cannot, or should not, be probed with a fixture probe. For example, if a probe and fixture wire on the clock pin of a device might alter the device's characteristics, you can specify the device pin as a **NO_PROBE** location. Note that you can also specify a node as **NO_PROBE**. You can specify **NO_PROBE** locations for the SimPlate fixture; Board Consultant creates the `board_xy` file.

Table 3-4 Probe access attributes (continued)

Attribute	Description
CRITICAL	This instructs the MPA stage of the fixture generation software to give special attention to nodes that are connected to critical device pins. MPA assigns resources to critical nodes before assigning resources to other nodes. This ensures that the physically shortest connections are applied to the resources specified as critical. MPA automatically determines the order in which to assign resources to nodes. Specifying critical pins overrides MPA's algorithm. There are only two cases where you should override MPA's algorithm by specifying critical device pins: • If you have a node that is used, but not specified, as a clock in a library test. • If you have a device to be tested that meets these criteria: - The device has edge-sensitive inputs. - The edge-sensitive inputs are not overdriven, or the device upstream from the edge-sensitive inputs is in a high-impedance state. The fixture generation software also assigns ground wires to the corresponding pin cards first to promote short ground wires on these pin cards. Board Consultant places the **CRITICAL** attribute next to the device pin in the `board_xy` file. The fixture generation software assigns the corresponding node as **CRITICAL** in the `fixture.o` file.

Table 3-5 Fixture access attributes

Attribute	Description
BOTTOM	Specifies that the location is accessible only from the bottom side of the board under test. If no attribute is specified, **BOTTOM** is assumed.
TOP	Specifies that the location is accessible only from the top side of the board under test. This is valid for express and cassette fixtures; top side probing with express fixtures requires custom modifications to the fixture hardware.
BOTH	Specifies that the location is accessible from the top and bottom sides of the board under test. This is valid for express and cassette fixtures; top side probing with express fixtures requires custom modifications to the fixture hardware.

Table 3-6 Manual access attributes

Attribute	Description
MANUAL	Indicates a location that can be probed with the guided-probe. If no attribute is specified, **MANUAL** is assumed.
NO_MANUAL	Indicates a location that cannot, or should not, be probed with the guided-probe. You can also specify a node as **NO_MANUAL**.

f Select the **Enter Internal Devices** action.

Use the displayed form to enter internal devices of clusters. Also enter the internal devices of device packages when you chose not to use Part Description Libraries.

g Select the **Enter Node** action.

Use the displayed form to enter or edit nodes. You can influence how the Probe Select, Module Pin Assignment (MPA), and Fixture Tooling (FXT) stages of the fixture generation software handle nodes by specifying node attributes.

The attributes that you can assign to nodes are grouped in three categories shown in **Table 3-7**. The Node Attributes are shown in **Table 3-8**, Manual Probe Attributes are shown in **Table 3-9**, and the Long Probe Attributes are shown in **Table 3-10**.

Table 3-7 Node attributes categories

Attribute	Description
Node Attributes	Attributes that apply to the entire node (all locations).
Manual Probe	Attributes that specify accessibility to the entire node (all locations) by the guided probe.
Long Probe	Attribute that specifies whether a long probe should be used on the location chosen for the node.

Table 3-8 Node attributes

CRITICAL	If a location (`device.pin`) is specified as **CRITICAL** during device entry, the associated node is automatically specified as **CRITICAL**. The **CRITICAL** attribute instructs the fixture generation software to handle the node differently from other nodes: • Probe Select tries to use short, 100-mil probes on critical nodes. Probe Select also tries to probe the critical device.pin, if one was declared in Board Consultant. • MPA assigns critical nodes before other types of nodes to promote short fixture wires on critical nodes. • Fixture Tooling specifies a short ground wire on the Pin Card associated with the **CRITICAL** node.
NO_PROBE	Indicates that the node cannot, or should not, be probed with a fixture probe. For example, an electronically sensitive node where fixture wiring might alter the node's characteristics, or a node that is not physically accessible with a fixture probe, should be specified as **NO_PROBE**. If you specify **NO_PROBE** nodes for the **SimPlate** fixture, Board Consultant creates a `board_xy` file. You can also specify a probing location as **NO_PROBE**.
NORMAL	Specifies that the fixture generation software should use its algorithm concerning this node. If no attribute is specified, **NORMAL** is assumed.
RESERVED_WIRING	Prevents additional wires or probes from being attached to the node.

Table 3-9 Manual probe attributes

Attribute	Description
MANUAL	Indicates that the node can be probed with the guided-probe. If no attribute is specified, **MANUAL** is assumed.
NO_MANUAL	Indicates that the node cannot, or should not, be probed with the guided-probe. You can also specify a location as **NO_MANUAL**.

Table 3-10 Long probe attributes

Attribute	Description
LONG_PROBE	Specifies that a long probe must be used on this node for dual-stage testing. If you do not specify a long probe, Probe Select assigns a short probe. Long probes are not allowed for the SimPlate fixture.
NORMAL	Specifies that a long probe is not used. If no attribute is specified, **NORMAL** is assumed.

View / Edit Test System Data

1 Select the View / Edit Test System Data block.

2 Select the Enter Power Node Information action.

3 Connect the DUT supplies in parallel.

4 Connect DUT supplies in series.

5 Select the Enter Fixed Node Information action.

6 Select the Enter Board-Level Disables/Conditions action.

7 Select the Enter IPG Global Options action.

8 Select the Enter Family Options action.

9 Select the Enter Fixture Options action.

10 Select the Enter GP Relay Connections action.

11 Select the Enter Board Keepout Areas action.

12 Select the Enter Groups action.

13 Select the Enter Extra Probing Locations action.

1 Select the View / Edit Test System Data block.

The data that you enter or edit under this selection is shown in **Table 3-11**.

Table 3-11 Data entered in the View / Edit Test System Data block

Data	Description
Power Node Information	Nodes that need to be connected to DUT power supplies and the ground node.
Fixed Node Information	Nodes that cannot be driven, such as a node connected to the ground node by a jumper. All Power nodes must also be entered as fixed nodes.
Board-Level Disables/Conditions	Information about devices and nodes which need to be disabled or pre-conditioned, but are not automatically handled by bus disabling or upstream conditioning.
IPG Global Options	Options that apply to the entire board.
Family Options and Card Preferences	Logic family values and which pin cards those families should use.

Table 3-11 Data entered in the View / Edit Test System Data block (continued)

Data	Description
Fixture Options	Options that apply to the test fixture. Note: These options are ignored if a valid fixture default file is created by the test developer, because it has higher precedence. See **Fixture Defaults File** on page 5-51 in the *Data Formats* documentation.
GP Relay Connections	For General Purpose relays on the Control Card or the AccessPlus Card.
Board Keepout Areas	Areas of the board where no probes or personality pins can be placed due to physical restrictions.
Groups	Instructions for twisted pair and coaxial wiring of paired nodes.
Extra Probing Locations	Extra locations to be partially drilled for future use.

2 Select the Enter Power Node Information action.

a Use the displayed form to list the power supply nodes and the ground node on the board to be tested.

b Specify the DUT power supply number and appropriate voltage and current limit values for each power node.

The fixture generation software uses this information to write the fixture wiring instructions for the DUT power supplies. IPG uses this information to find device pins that are tied high (to VCC), or low (to ground), when generating digital device tests.

c Enter the power nodes also in the **Fixed Node Entry** form.

3 Connect the DUT supplies in parallel.

You can connect DUT power supply outputs in parallel to supply more current to the board under test. You need to be aware of the type of power supplies in your test system and to which ASRU channels they are connected because there are restrictions for which supplies can be connected in parallel. You can connect:

- Low voltage outputs to low voltage outputs, and high voltage outputs to high voltage outputs only.

CAUTION

⚠ Never connect a low voltage output to a high voltage output.

- Outputs that use ASRU channels 1-4 with other outputs that use ASRU channels 1-4, and outputs that use ASRU channels 5 and 6 with other outputs that use ASRU channels 5 and 6.

CAUTION

⚠ Never connect an output that uses ASRU channels 1-4 with an output that uses ASRU channel 5 or 6!

- Up to eight outputs of Agilent 6621 supplies that use ASRU channels 1-4.
- Up to four outputs of 6621 supplies that use ASRU channels 5 and 6.
- Up to eight low voltage outputs of Agilent 6624 supplies that use ASRU channels 1-4.
- Up to eight high voltage outputs of 6624 supplies that use ASRU channels 1-4.
- Up to four low voltage outputs of 6624 supplies that use ASRU channels 5 and 6.
- Up to four high voltage outputs of 6624 supplies that use ASRU channels 5 and 6.
- Only two outputs of Agilent 6634 supplies.
- Only two outputs of Agilent 6642 supplies.

NOTE

See **Configuring DUT Power Supplies for Backward Compatibility** on page 5-25 in *Cards In The Testhead* for a description of the DUT power supplies.

a To specify parallel DUT power supplies, enter each supply in the **Power Node Entry** form.

b Specify the same node name for each supply.

c Specify the needed voltage for one of the supplies.

d Specify the needed voltage plus one percent for each of the other supplies.

e Specify the current for each supply as shown in **Table 3-12**.

Table 3-12 Power supply settings

Setting	Specification	Example
for the supply set to the lowest voltage value	Half the maximum current capability of the supply plus half the expected current swing of the board under test; we recommend that you increase the value to the next integer.	The board under test requires 10 to 12 amps (swing of two amps) and you are using three supplies capable of five amps each: Half of its capability (5 / 2 or 2.5 amp) plus half of the expected swing (2 / 2 or 1 amp) = 3.5 amps; round up to 4 amps.
for the other supplies	The maximum requirement of the board under test minus half the capability of the first supply divided by the number of other supplies.	The board under test requires 10 to 12 amps (swing of two amps) and you are using three supplies capable of five amps each: The total requirement (12 amps) minus half of the first supply’s capability (5 / 2 or 2.5 amps) = 9.5 amps divided by the number of other supplies (9.5 / 2) = 4.8 amps each.

4 Connect DUT supplies in series.

You can connect DUT power supplies in series for higher voltage values than are possible with just one supply. Connecting the supplies in series is a manual task.

WARNING

⚠ Never apply more than 100 volts to the board under test with the DUT power supplies and internal reed relays because this could damage the test system and expose the operator to hazardous voltages.

WARNING

⚠ Do not connect Agilent 6634 or Agilent 6642 power supplies in series.

a Enter the DUT supplies in the **Power Node Entry** form.

b Specify the first DUT supply as connected to the power node with a positive voltage value equal to half the desired voltage.

c Specify the second supply as connected to an imaginary node with a negative voltage value equal to half the desired voltage.

Make sure that the imaginary node is associated with a device and an X-Y location.

5 Select the Enter Fixed Node Information action.

Use the displayed form to list the fixed nodes and the power nodes on the board.

Fixed nodes are any nodes that cannot or should not be driven during testing, such as a node connected to a power node through a jumper or very small resistor. Fixed nodes include any node that is connected to a power pin of a device. Each fixed node must have a logic level (0, 1, or X), specified. IPG uses this information when analyzing the digital library tests.

6 Select the Enter Board-Level Disables/Conditions action.

Use the displayed form to define the board level disabling and conditioning.

NOTE

For a complete description of disabling and conditioning see Chapter 2, **Vector Control Language (VCL)** in *Test Methods: Digital*.

7 Select the Enter IPG Global Options action.

Use the displayed form to specify the IPG Global Options.

The IPG Global Options affect how IPG writes the device tests. The IPG Global Options are shown in **Table 3-13** on page 3-35.

Table 3-13 IPG global options

Option	Description
Tolerance Multiplier	Specifies how accurately a device should be tested. This can be from 0.1 to 10. The default is 5. IPG uses this number to determine how accurate a test to write. The smaller the value of the tolerance multiplier, the more accurate the test. More accurate tests require more test time and resources.
Diode Current	Specifies the maximum current (in amps) to be used in a diode test. The default is 5mA. IPG uses this value to generate diode tests.
Fuse Threshold	Specifies the resistance value (in ohms) to be used to determine the presence of a fuse. The default is 10 ohms. If the resistance of the fuse is less than or equal to the threshold, the fuse is considered present. IPG uses this value to generate fuse tests.
Zener Current	Specifies the maximum current (in amps) to be used in a zener test. The default is 10mA.
Adjust	Specifies the type of adjustment to be used when testing adjustable devices. This can be **NONE**, **ACCURATE**, or **FAST**. The default is **ACCURATE**. If you specify **NONE**, the device is tested without allowing the operator to adjust the part. If you specify **ACCURATE**, the operator is prompted to adjust the device every time it is tested. If you specify **FAST**, the operator is prompted to adjust the device only if it does not measure within tolerance.
Remote Sensing	Indicates if remote sensing should be used. This can be **ON** or **OFF**. The default is ON; use remote sensing. IPG uses this value to determine if it should use remote sensing if needed, or not at all. Remote sensing provides for more accurate tests; however, it requires more test time and resources.
Upstream Disable	Specifies if IPG should try to disable upstream devices. This can be **OFF** or **ON**. The default is **OFF**.

Table 3-13 IPG global options (continued)

Option	Description
Capacitor Compensation	Specifies if capacitor compensation is to be used. This can be **ON** or **OFF**. The default is **OFF**; do not use capacitor compensation. IPG uses this value to determine if it should invoke capacitance compensation on small capacitors.
Common Lead Resistance	Specifies the typical resistance (in ohms) of the fixture probe and board trace to the device under test. The range of acceptable values is 0.1 ohms to 100 ohms. The default is 0.5 ohms. IPG uses this value in generating device tests.
Common Lead Inductance	Specifies the typical inductance (in Henries) of the fixture probe and board trace to the device under test. The range of acceptable values is 0.1nH to 1mH. The default is 1uH. IPG uses this value in generating device tests.
IPG Digital Resistance Threshold	Specifies the maximum resistance between device pins for IPG to treat them as **tied** pins (connected together).
Precondition Levels	Sets the number of levels of upstream devices that IPG should try to disable or condition. See Chapter 2, **Vector Control Language (VCL)** in *Test Methods: Digital* for complete description of preconditioning.
Additional Board Voltage	Specifies the MAXIMUM applied or generated voltage on the DUT from a source other than the Agilent 3070 DUT power supplies. This includes on board power supplies, external power sources, fixture electronics, and dc-to-dc converters. This value is added to the sum of all non-parallel 3070 DUT power supply voltages specified in the board file for this board and is used by the capacitor discharge routines to determine which capacitors need to be discharged. The entered value must be an integer between zero and 100. The default is 0. If the sum of all non-parallel power supplies plus that Additional Board Voltage is greater than 100 VDC a warning will be issued during board compile.

Table 3-13 IPG global options (continued)

Option	Description
Use Agilent Drive-Thru Test	When Drive-Thru Test software is installed and enabled, this option is set to **YES**. Otherwise, the default setting is **NO**.
Agilent Drive-Thru Impedance Threshold	Set the maximum impedance of devices that can be tested using Drive-Thru Test: resistors, capacitors, and inductors. The default setting is 10K ohms. This option is only available when Drive-Thru Test is enabled.
Boundary-Scan Disable	Determines whether boundary-scan disable is **ON** or **OFF**. The default is **OFF**; the scan is enabled.
Boundary-Scan Overdrive	Specifies that the shortest possible chain be used for the connect test. This can be **ON** or **OFF**. The default is **OFF**; do not require the use of the shortest possible chain.
Boundary-Scan Chain Override	Allows manual additions of chain descriptions to the **board** file. Also includes chain descriptions in the listing of the `board` file. This can be **ON** or **OFF**. The default is **OFF**; do not allow the manual additions.
Ground Bounce Suppression	Determines whether Ground Bounce Suppression is **ON** or **OFF**. The default is **OFF**. When changed, IPG Test Consultant regenerates all connect and interconnect tests.
Powered Shorts Shorting Radii	Set the Radius in mils to identify the powered shorts adjacencies. The default setting is 100 mils. When changed, IPG Test Consultant regenerates all connect and interconnect tests.
Test Strategy	Specifies the test strategy to be used. This can be **COMBINATIONAL** or **EDGE CONNECTOR ONLY**.
Board Heading	A string variable to be included at the top of any messages printed by the device that is specified by the `report is` statement. You can set this to any message that you want to be printed at the top of failure reports.

Table 3-13 IPG global options (continued)

Option	Description
Unconnected Pin	A name that indicates device pins which are intentionally left unconnected.
Board File List Format	Specifies the format of the `board` file. It can be **devices**, **nodes**, or **both**.

8 Select the Enter Family Options action.

Use the displayed form to specify the family options and card preferences.

The family options determine the drive levels used to test digital devices; determine what level of output is considered a low or a high; and specify the edge speed of the device, the logic level of open inputs, and the load to be placed on outputs. The family options are shown in **Table 3-14**.

Table 3-14 Family options

Option	Description
Family-id	Specifies the name of the logic family being defined. The name must correspond to a logic family named in a device library.
Drive High	Specifies the voltage used to drive a node to a logic 1. This can be a value from -3.5 volts to 5 volts. The defaults are: CMOS — 4 TTL — 3.5 ECL — -0.5.
Drive Low	Specifies the voltage used to drive a node to a logic 0. This can be a value from -3.5 volts to 5 volts. The defaults are: CMOS — 0.8 TTL — 0.2 ECL — -2.7.
Receive High	Specifies the lowest voltage that can be considered a logic 1. This can be a value from -3.5 volts to 5 volts. The defaults are: CMOS — 3.6 TTL — 2 ECL — -1.2.
Receive Low	Specifies the highest voltage that can be considered a logic 0. This can be a value from -3.5 volts to 5 volts. The defaults are: CMOS — 1.3 TTL — 0.8 ECL — -1.5.

Table 3-14 Family options (continued)

Option	Description
Slew Rate	Specifies the edge transition speed in volts per microsecond rounded to the nearest 25. The range of edge speed is 25 to 275 volts/uSec. The defaults are: CMOS — 50 TTL — 100 ECL — 150.
Open Input Default	Specifies the logic level that an unconnected input pin assumes. This can be: 0, 1, or X. The defaults are: CMOS — X TTL — X ECL — 0.
Load	Specifies the load to be placed on outputs. This can be **NONE**, **UP**, or **DOWN**. The defaults are: CMOS — NONE TTL — UP ECL — DOWN.

Card preference specifies the association of types of pin cards to logic families, and assigns the priority that determines how the resources on the pin cards are allocated by the fixture generation software.

9 **Select the Enter Fixture Options action.**

Use the displayed form to specify the fixture options. The fixture options determine how the fixture generation software creates the fixture files and reports that are used to build the test fixture. The fixture options are shown in **Table 3-15**.

Table 3-15 Fixture options

Option	Description
Fixture Type	Specifies the type of fixture to be used. The type can be: **Simplate**, **Express**, **No-Wire**, **Cassette**, **JOT**, **QuickPress**, **XG-50**, or **XG-50 Cassette**. The default is **Express**. The fixture generation software uses this value to determine which type of fixture files and reports to generate. If you are using an express fixture initially and want to convert it to a cassette fixture in the future, specify the fixture type as **Cassette** initially.
Fixture Size	Specifies the size of the fixture. The size can be **BANK1**, **BANK2**, or **FULL**. The default is **FULL**. The fixture generation software uses this value to determine which testhead modules are covered by the fixture.
Electrical Top Probes	Determines if the fixture generation software is allowed to use probe locations that are accessible only from the top side of the board under test. This value can be **ON** or **OFF**. The default is **OFF**: do not allow the use of probe locations that are accessible only from the top of the board. You can allow top probes for any type of test fixture; however, using top probes on fixtures other than cassette fixtures, requires custom mechanical modifications to the fixture.
Heavy Probe Force	Specifies the pressure, in ounces, exerted by a heavy weight probe. The default is eight. The fixture generation software uses this value, along with the density threshold, to determine the maximum number of probes per square inch allowed on the board under test.
Light Probe Force	Specifies the pressure, in ounces, exerted by a light weight probe. The default is four. The fixture generation software uses this value, along with the density threshold, to determine the maximum number of probes per square inch allowed on the board under test.

Table 3-15 Fixture options (continued)

Option	Description
Mechanical Probe Density	Specifies the maximum probe force, in ounces per square inch, that can be exerted on the board under test with a mechanical (cassette) type fixture. The default is 800. The fixture generation software uses this value, along with the probe force, to determine the maximum number of probes per square inch allowed.
Vacuum Probe Density	Specifies the maximum probe force, in ounces per square inch, that can be exerted on the board under test with a vacuum (express) type fixture. The default is 104. The fixture generation software uses this value, along with the probe force, to determine the maximum number of probes per square inch allowed.

Table 3-15 Fixture options (continued)

Option	Description

Autofile	Specifies a numeric code to automatically identify the fixture that is loaded on the testhead. This can be a number from 11 to 4094. If no autofile is specified, the fixture generation software automatically assigns the first available value starting at 4094 and decreasing to 11. The fixture generation software uses this value to add the necessary wiring to the fixture. The fixture generation software also creates a corresponding file under `$AGILENT3070_ROOT/autofile/<autofile value>`. This file contains the name of the board and the absolute path to the board directory. For example:
	`ps_board`
	`$AGILENT3070_ROOT/boards/ps_board`
	For multiple-board fixtures, you need to edit the autofile file to include a listing for each board on the fixture. For example:
	`board_1`
	`$AGILENT3070_ROOT/boards/board_1`
	`board_2`
	`$AGILENT3070_ROOT/boards/board_1`
	`board_3`
	`$AGILENT3070_ROOT/boards/board_1`
	Note that QuickPress uses two transfer probes to change the Autofile number only when the top plate is engaged. This allows the software to determine when to proceed with the board test.

Table 3-15 Fixture options (continued)

Option	Description
WireWrapping	Specifies the method to be used to wire the fixture. This can be **MANUAL**, **SEMI-AUTO**, **WIRELESS**, or **AUTO**. The default is **MANUAL**. This value is used by the fixture generation software to determine if the wiring report should be formatted for manual (human readable) or automatic (machine readable) wiring. Note that **SEMI-AUTO** and **AUTO** are the same; if you specify **SEMI-AUTO**, the `board` and `fixture.o` files specify **AUTO**. With the **WIRELESS** option selected, probe select will not block a personality pin that is close to a DUT probe, except for those near tooling holes. The fixture tooling process will generate a new output file, named `nets`, in the fixture directory. This file is designed to be translated into various PCB auto-router software formats.
Metric Units	Specifies whether the fixture generation software should write its reports in metric or English units. This value can be **ON** or **OFF**. The default is **OFF**; write in English units.

10 Select the Enter GP Relay Connections action.

Use the displayed form to specify the GP relay connections.

General Purpose (GP) relays can be used to hold an enable line high or low for several tests, or connect a load to a motor drive circuit.

Eight GP relays are available on each Control Card in the testhead. The optional AccessPlus Card can provide up to 24 more GP relays.

GP relays on the Control Card can be opened and closed during controller loops; GP relays on the AccessPlus Card cannot.

The fixture generation software automatically includes the GP relay wiring instructions in the fixture files and reports.

11 Select the Enter Board Keepout Areas action.

Use the displayed form to specify board keepout areas.

A board keepout area is typically one that has physical restrictions that prevent placing probes in an area of the PC board.

A keepout area must be specified as a series of at least three points that define a polygon.

a Enter the coordinates of the corners of the keepout area starting at one point.

b Proceed around the area in either a clockwise or counter-clockwise direction, continuing to enter coordinates.

c Specify the keepout area to be on the bottom or top side of the board, or on both sides.

If you do not specify which side of the board, the software assumes the bottom side.

A board keepout area is relative to the board origin and moves with the board if the board placement is changed or if the board is rotated.

12 Select the Enter Groups action.

Use the displayed form to specify groups.

a If your board test employs external instrumentation, use coaxial cable or twisted-pair wiring to maintain the signal integrity.

b Specify the appropriate nodes as GROUPS. The device pins that correspond to these nodes must be connected to external ports or serial ports in a device test; MPA assigns AccessPlus or Serial Card resources to these nodes so that they can be wired with coaxial or twisted pair wiring.

Groups must consist of two device pins that are accessible from the bottom of the board. The `board_xy` compiler warns you if the distance between the device pins specified in a group is too great for the wire type of that group; however, the wire is assigned and an error is not generated. The maximum recommended separation for a COAX group is 0.5 inches; the maximum recommended separation for a TWISTED group is 1.0 inch.

13 Select the Enter Extra Probing Locations action.

Use the displayed form to specify extra probing locations.

Extra probing locations are drilled locations, ready for future use. These locations are not considered by Probe Select, but are passed on to the `fixture.o` file as alternates. The alternates in the `fixture.o` file are considered for probing locations when the fixture generation software is re-run.

The extra probe location attributes are grouped into two categories as shown in **Table 3-16**. The Fixture Access attributes are described further in **Table 3-17**, and the Manual Access attributes are presented in **Table 3-18**.

Table 3-16 Extra probe location attributes

Attribute	Description
Fixture Access	Attributes that specify the accessibility of the probe locations by the fixture probes.
Manual Access	Attributes that specify the accessibility of the probe locations by the guided probe.

Table 3-17 Fixture access attributes

Attribute	Description
BOTTOM	Specifies that the location is accessible only from the bottom side of the board under test. If no attribute is specified, **BOTTOM** is assumed.
TOP	Specifies that the location is accessible only from the top side of the board under test. This is valid for express and cassette fixtures; top side probing with express fixtures requires custom modifications to the fixture hardware.
BOTH	Specifies that the location is accessible from the top and bottom sides of the board under test. This is valid for express and cassette fixtures; top side probing with express fixtures requires custom modifications to the fixture hardware.

Table 3-18 Manual access attributes

Attribute	Description
MANUAL	Indicates a location that can be probed with the guided-probe. If no attribute is specified, **MANUAL** is assumed.
NO_MANUAL	Indicates a location that cannot, or should not, be probed with the guided-probe.

Create the Board Configuration File

Ensure that a board configuration file, `config`, is in your local board directory. This board configuration file describes the hardware resources necessary to test your board.

1 Start Board Consultant.

2 Select the View / Edit Test System Data block.

3 Select the Compile / Verify block.

4 Select the Display config File Instructions action.

5 Copy the Standard Configuration (config) file.

6 Edit the standard configuration file.

7 Target another system model.

8 Enable Optional Features.

9 Compile the board configuration file.

10 Exit the BT-BASIC window.

11 Verify data and configuration.

1 Start Board Consultant.

2 Select the View / Edit Test System Data block.

3 Select the Compile / Verify block.

4 Select the Display config File Instructions action.

5 Copy the Standard Configuration (config) file.

The easiest way to create a board configuration file is to copy the standard configuration file to your local board directory. You might need to edit the file after copying it to your local board directory.

a Open a BT-BASIC window with the **Work** menu.

b `msi` to your local board directory.

c Use the copy (BT-BASIC) statement to copy the standard configuration file to your local board directory.

```
copy btgetenv$("AGILENT3070_ROOT") &
     "/standard/config" to "config"
```

6 Edit the standard configuration file.

a If the `config` file is not accurate for your test and testhead, edit the `config` file in your local board directory.

```
load "config"
```

b Specify the resources necessary to test the board.

- Specify the mandatory cards, such as the ASRU Card and the Control Card.

- State the proper number of pin cards. The test needs one brc available for every node on the board. You should include about 10% additional brc's.
- If you are developing a board test to be used on more than one testhead with different resources, the configuration file should state resources in locations (slot numbers) that are common to all appropriate testheads.
- For single-bank fixtures include only the resources in the appropriate bank; do not include the resources from the unused bank.

7 Target another system model.

Use the `target` statement to develop a board test and fixture for a system other than the model you are using. For instance, you can develop a board test and fixture on an Agilent 3075 to be used on an Agilent 3073.

8 Enable Optional Features.

Use the `enable` statement in the board-level configuration file `config` to enable optional features such as PanelTest. If you are targeting another 3070 Series 3 system, be sure to enable only those features available in the target system.

9 Compile the board configuration file.

Generate the object version of the board configuration file (`config.o`). You can select the **Display Config File Instructions** action from below the Board Consultant flow chart to display instructions on compiling the `config` file.

a Load the `config` file into the BT-BASIC workspace (if it is not there).

b Compile the file to generate the object version.

```
compile "config"
```

10 Exit the BT-BASIC window.

The Board Consultant window returns.

11 Verify data and configuration.

a Perform these actions shown in **Table 3-19** and make any necessary changes before continuing the test development process.

b If necessary, print the results of these checks:

- Select **Print...** from the **File** menu, or
- Select the **Testability Report** block to generate the `testability.rpt` file. You can then copy the `testability.rpt` file to the printer.

Table 3-19 Actions in the Compile / Verify block

Action	Description
Verify Fixture Type	Determines if the `config` file and fixture type are consistent.
Verify Configuration Size	Determines if there are sufficient resources declared in the board configuration file for the number of nodes on the board to be tested.
Verify Node Probing	Checks for a probing location on every node.
Verify Power Probing	Determines if there are sufficient probing locations on power nodes.
Verify Ground Probing	Determines if there are sufficient probing locations on ground nodes.

Enter Library Data

Provide the system with information about the libraries needed to test the devices on the board. The information includes the pathnames to the library directories. Also, create any necessary part description libraries and custom library tests. This task can be performed any time after creating the board directory and before compiling the `board` and `board_xy` files.

View / Edit Library Data

1 **Select the View / Edit Library Data block.**

2 **Select the Compile / Verify block.**

3 **Select the Enter Library Paths action.**

4 **Return to the View / Edit Library Data block.**

5 **Select Close to remove the Library Paths Entry Form.**

6 **Close the current instruction window.**

7 **Create Part Description Libraries.**

8 **Create custom library tests.**

9 **Compile and verify library data.**

---

1 **Select the View / Edit Library Data block.**

---

2 **Select the Compile / Verify block.**

a Determine which library directories are needed and how many tests are used from each directory.

Select the last three actions for this block. They are shown in **Table 3-20** on page 3-50.

Table 3-20 Actions in the **Verify** pull-down menu

Action	Description
Enter Library Paths	Displays a list of part numbers and devices that use library tests from the specified library directory.

Table 3-20 Actions in the **Verify** pull-down menu (continued)

Action	Description
Show Library for Device Designator	Displays the path to the library test object file for the specified device.
Show Library for Part Number	Displays the path to the library test object file for the specified part number.

3 Select the Enter Library Paths action.

a Use the displayed entry form to specify the library directory paths.

The library paths list is a list of directories that contain library test files. IPG searches these directories to find library tests for the devices to be tested. IPG searches the directories in the order listed.

- You should list custom library directories first to guarantee that they override any standard libraries.
- You should remove the paths to unused library directories so IPG does not use unnecessary time searching those directories.

4 Return to the View / Edit Library Data block.

Enter or edit the library paths.The library option defaults are:

- `$AGILENT3070_ROOT/library/ttl`
- `$AGILENT3070_ROOT/library/lsi`
- `$AGILENT3070_ROOT/library/cmos`
- `$AGILENT3070_ROOT/library/ecl`
- `$AGILENT3070_ROOT/library/setup`
- `$AGILENT3070_ROOT/library/linear`
- `$AGILENT3070_ROOT/library/interface`

Board Consultant places this information in the `board` file.

5 Select Close to remove the Library Paths Entry Form.

You can select the additional actions in **Table 3-21** to display instructions for creating libraries and safeguard files

.

Table 3-21 Additional actions in the View / Edit Library Data block

Action	Description
Display Device Library Instructions	Displays instructions for creating and compiling device library files.
Display Part Library Instructions	Displays instructions for creating part libraries.
Display Safeguard File Instructions	Displays instructions for creating and compiling safeguard files.

6 Close the current instruction window.

This returns you to the flow chart.

7 Create Part Description Libraries.

When there are more than two devices within one physical part, use the part description library to describe the device. Each description of a part is written in Part Description Language (PDL) and resides in an individual file.

NOTE

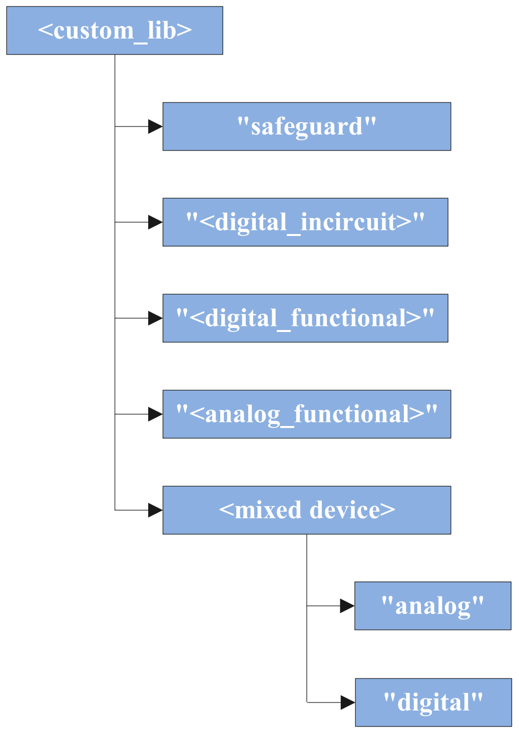
PDL is explained in Chapter 8, **Part Description Language** in *Data Formats*.

8 Create custom library tests.

You need to provide tests for any devices to be tested that do not have library tests in the 3070 Series 3 standard library directories. You also need a setup-only library test for Boundary Scan tests even if you have a BSDL file.

Figure 3-5 on page 3-53 shows the custom library directory structure. Note that file names are in quotes (“file_name”) and directory names are not (dir_name). The file and directory names in angle brackets (<>) are arbitrary—you can choose the names you want; the names without angle brackets must be as shown in the figure.

Figure 3-5 Custom library directory structure



a Perform the **Verify Missing Libraries** testability check:

- Select **View / Edit Library Data** block.
- Select the **Compile / Verify** block.
- Select the appropriate action below the flow chart.

This testability check tells you which devices (by part number) do not have a library test source file. It also tells you which devices (by reference designator) need that library. From this information, you can determine how many libraries need to be written, and which library tests are needed by which devices.

b Use the `find library` statement to search the standard libraries for a specified device test. Create the custom library directory if it does not exist, and be sure it is listed in the Library Paths Entry Form.

c For mixed tests, also create the mixed test directory. See **Figure 3-5** on page 3-53.

d Write the custom library tests or copy and modify standard library tests.

See **Creating Custom Library Tests: Reference** on page 3-71 for more information.

e Exit the BT-BASIC window to return to the flow chart.

9 **Compile and verify library data.**

a Select the **View / Edit Library Data** block.

b Select the **Compile / Verify** block.

c Select the appropriate actions to save the `board` file and to compile the library tests and the `Agilent safeguard` file. These actions are listed in **Table 3-22** on page 3-54.

d If necessary, print the results of these checks:

- Select **Print...** from the **File** menu; or,
- Select the Testability Report block to generate the `testability.rpt` file. You can then copy the `testability.rpt` file to the printer.

e Close the current window to return to the flow chart.

Table 3-22 Actions in the Compile/Verify block

Action	Description
Save Board Files	Re-stores the `board` file because the library paths were added.
Compile Modified Libraries	Compiles all new and modified library test files.
Compile Modified Safeguard Files	Compiles all new and modified safeguard files.

Table 3-22 Actions in the Compile/Verify block (continued)

Action	Description
Verify Missing Libraries	Displays a list of missing library source files.
Verify Disable Methods Exist	Determines if there is a disable method for every bussed output pin and every bidirectional pin.
Verify Disable Nodes Usable	Determines if the pins used in the disable methods are usable—probed and not tied high or low.
Verify Tied Nodes Data	Displays a list of device pins that are tied high or low.
Verify Boundary-Scan Chains	Displays a list of pin devices that will be tested using Agilent Boundary-Scan.
Show Boundary-Scan Chains	Displays a list of Boundary-Scan chains on the left-hand panel. Double-clicking on a chain will show the devices in the chain on the right-hand panel of the window. Selecting a device will show you details about that device including pin numbers, the location of overrides, node names, and BDSL information.
Show Devices In Library Directory	Displays a list of part numbers and devices that use library tests from the specified library directory.
Show Library for Device Designator	Displays the path to the library test object file for the specified device.
Show Library for Part Number	Displays the path to the library test object file for the specified part number.

Compile the Board and X-Y Data Files

1 **Select the Final Compile / Verify Block.**

2 **Select the Save the Board Files action.**

This saves any changes to the `board` and `board_xy` files.

3 **Select the Compile Board File action.**

Before you can compile the `board` file, all library tests must be available and you must have compiled the `config` file.

You now have a `board.o` file in your local directory.

Compiling the `board` file also creates a `safeguard` file in your local board directory. This file is created from the individual `safeguard` files in the library directories. This is the board-level safeguard; it references all digital tests and the digital portion of mixed tests (which are being created from libraries—not custom executable tests) for the board under test.

4 **Select the Compile board_xy File action.**

You now have a `board_xy.o` file in your local directory.

Examine the Files

1 **Select the Testability Report block.**

2 **Print the Testability Report.**

3 **Verify the board_xy.o file.**

4 **Exit Board Consultant.**

1 Select the Testability Report block.

You can use the testability report to correct any problems in your board test before continuing. You can also use the report to provide information about the testability of the board to the design department.

The testability report has two sections, summary and details. A complete testability report includes the following topics in both the summary and details sections:

- a list of files from which the testability information was derived;
- Missing Library Results - library devices that have no library sources;
- Agilent Polarity Check Test Results - capacitors that are tested using Polarity Check;
- Agilent TestJet Results - devices that are tested using Agilent TestJet;
- Keepouts Analysis Results - whether there are at least as many bottom-side keepout areas as there are bottom side devices being tested with Polarity Check and Agilent TestJet;
- Agilent Boundary Scan Results - devices that are tested with Boundary Scan;
- Tied Pins Results - device pins that are tied high, low, or connected to a fixed node;
- Disable Methods Results - device pins that need to be disabled but have no disable method in the library test;
- Useful Disabling Nodes Results - device pins that are needed for disabling but are tied high or low;
- Node Probing Access Results - nodes that have no probing access;
- Power Node Access Results - power nodes that have insufficient probing access;
- Ground Node Access Results - whether the ground node has sufficient probing access;
- Config and Fixture Results - whether the configuration, fixture size, and fixture type are consistent;
- Configuration Size Results - whether the configuration is sufficient to test the board;

- Device Test Results - devices that have no test, limited test, commented test, etc.;
- Device Disable Results - devices that have disable errors and devices that are causing those errors;
- Safeguard Analysis Results - devices that have safeguard inhibits and devices that are causing those inhibits.

The report is placed in a file called `testability.rpt`.

2 Print the Testability Report.

Use the `list source` statement to copy the testability.rpt file to the printer.

3 Verify the board_xy.o file.

It is essential that the `board_xy` file be complete and accurate before you proceed further in the process because it determines how the fixture is built and affects the device tests.

NOTE

The 3070 Series 3 provides a Plot Generator (described in Chapter 5, **The Plot Generator** in *Test Development Tools*) that you can use as a final verification of the `board_xy` file.

a Use the `generate plot` statement to produce a plot file of the `board_xy` file.

The plot file is called `board_xy.p` (also `board_xytop.p` for the top plate of the cassette fixtures) and is placed in the local board directory. You can copy the plot file over a plotter to generate a plot of the X-Y data.

b Plot the `board_xy.p` file on a clear piece of mylar with a 1:1 ratio. Place the mylar over the blank board to see how the holes and probe locations match the device holes or pads on the PC board.

4 Exit Board Consultant.

Summary

You have now created the board directory, the board configuration file, the board data file, and X-Y data file.

You also created any necessary custom library tests including part description libraries.

File Structure

Figure 3-6 on page 3-60 shows the structure of your local board directory after creating Board Information. The directories created are shown in **Table 3-23** on page 3-61. The files created are shown in **Table 3-24** on page 3-61.

Figure 3-6 File structure after creating board information

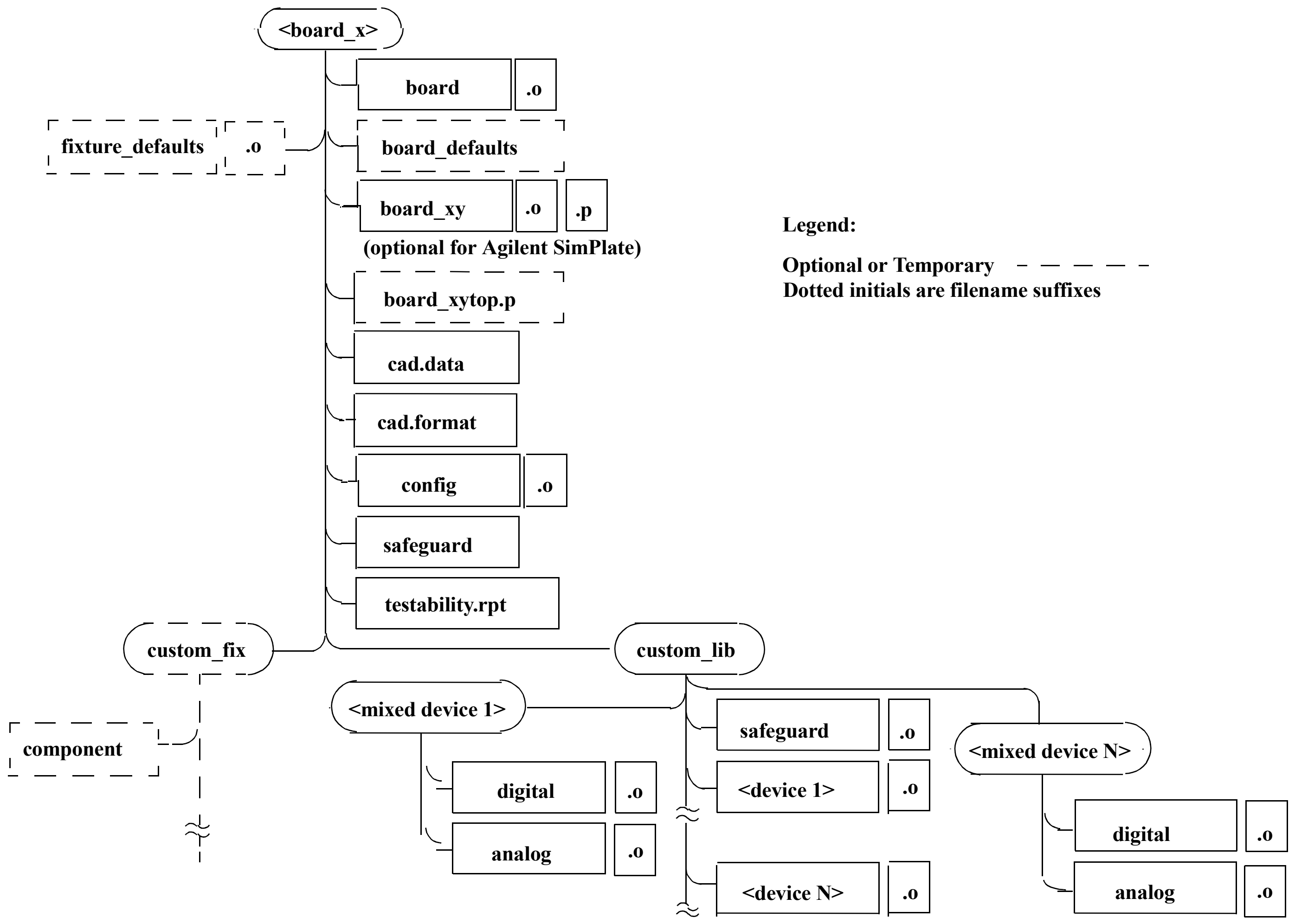


Table 3-23 Directories created while creating board information

Directory	Description
<board>	The local board directory.
custom_lib	Contains your custom library tests and `safeguard` file if needed. If you have no custom tests, this directory does not exist.
custom_lib/<mixed device>	Contains the library files for a mixed test. The library sources are `digital` and `analog`. There is one directory for each mixed test.
custom_fix	Contains all fixture component files

Table 3-24 Files created while creating board information

Directory	Description
<board>/config	The board configuration source file that describes the hardware necessary to test the board.
<board>/config.o	The compiled (object) version of the board configuration file.
<board>/board	The source file that contains topology information for the board under test.
<board>/board.o	The compiled (object) version of the `board` file.
<board>/board_xy	Contains the X-Y data that physically describes the board for which the test is being developed. This file is optional for the SimPlate fixture.
<board>/board_xy.o	The compiled object file of the `board_xy` file.

Table 3-24 Files created while creating board information (continued)

Directory	Description
<board>/board_defaults	Provides any missing data for the `board` file. This file is optional, Board Consultant uses the `$AGILENT3070_ROOT/standard/board_defaults` if there is no `board_defaults` file in the local board directory.
<board>/board_xy.p	The plot file of the `board_xy` file.
<board>/board_xytop.p	The plot file of the `board_xy` file for the top probes in the cassette fixtures.
<board>/testability.rpt	A report file that provides information about the testability of the board. This report includes information about the device tests, node accessibility, and configuration size.
custom_lib/safeguard	The `safeguard` file for each custom library directory that contains a digital test or a digital portion of a mixed test. If you have no custom digital or mixed library tests, this file does not exist.
custom_lib/safeguard.o	The compiled object version of the library `safeguard` file.
custom_lib/<device>	The source file for each custom library test. Note that a custom library for a mixed test consists of a directory named for the part number of the device and includes two files—`analog` and `digital`. If you have no custom library tests, these library files do not exist.
custom_lib/<device>.o	The compiled library object of each custom library test source file. For a mixed test there are two files in the mixed library test directory—`analog.o` and `digital.o`.
custom_lib/<mixed device>/digital	The source file for the digital portion of a mixed library test.
custom_lib/<mixed device>/digital.o	The compiled library object version of the `digital` file for a mixed library test.

Table 3-24 Files created while creating board information (continued)

Directory	Description
custom_lib/<mixed device>/analog	The source file for the analog portion of a mixed library test.
custom_lib/<mixed device>/analog.o	The compiled library object version of the `analog` file for a mixed library test.
<board>/safeguard	The board-level `safeguard` file. This is created when you compile the `board` file. This file is made from all the individual library directory `safeguard` files.

Using the Part Description Editor: Reference

Working with the Forms

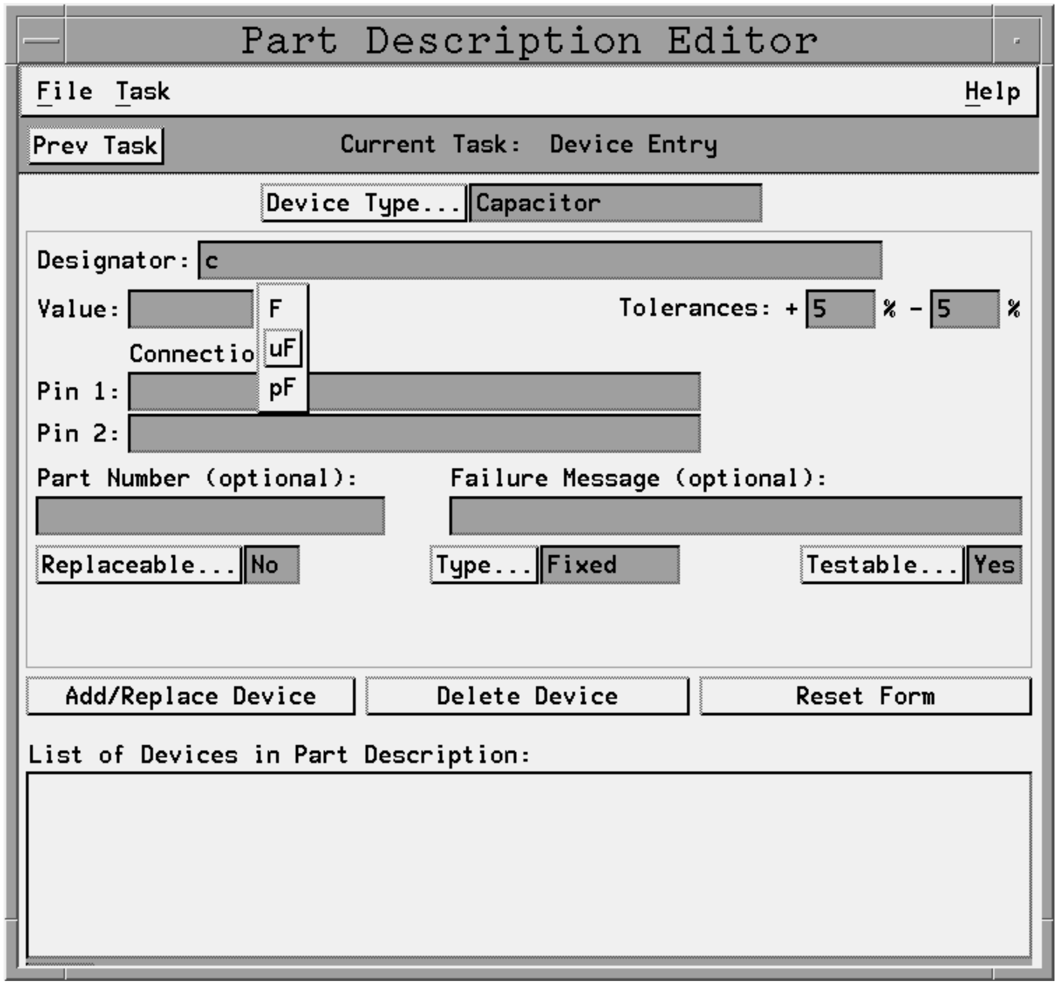
To move among the data entry fields in a form, you can:

- Press Tab or Shift-Tab (back-Tab).
- Click the mouse's left button on a data entry field.
- Enter a value, press Return, and automatically move to the next field.

When you select a field, it becomes highlighted and the typing cursor moves into it.

- The text box below List of Devices in Part Description: shows the designators of devices currently in the part description library. Click the mouse's left button on any of these designators to automatically open the correct form and go directly to editing the entry for that device.
- (Only in the **Initialization** form) Internal Nodes are updated to the appropriate number of rows only after you enter a value for Number of Internal Nodes and press **Return**.
- (Only in the **Connector Entry** form, the **Pin Library Entry** form, or the Switch Entry form) Pins and Connections are updated to the appropriate number of rows only after you enter a value for Number of Connections and press **Return**.
- Most device entry forms let you specify some options by choosing pre-defined values from pull-down menus. For example, using the mouse to push the button containing F (farads) in the **Capacitor Entry Form** opens a pull-down menu from which you can select units of measure, as in **Figure 3-7** on page 3-65. Use the mouse to drag the selection bar to your choice, then release the button. The units change to the chosen value.

Figure 3-7 A sample Capacitor entry form



Part Description Editor

Current Task: Device Entry

Device Type... Capacitor

Designator: c

Value: F uF pF

Tolerances: + 5 % - 5 %

Connectio

Pin 1:

Pin 2:

Part Number (optional):

Failure Message (optional):

Replaceable... No

Type... Fixed

Testable... Yes

List of Devices in Part Description:

- The buttons that appear near the bottom of each device entry form are in Table 3-26 on page 3-69.
- Clicking the mouse's left button on a designator below **List of Devices** in Part Description: in a device entry form automatically opens the appropriate form and displays the description of the chosen device. You can view the description, modify it, or replace it as needed.
- Clicking the mouse's left button on Prev Task in a device entry form returns you to the **Initialization** form.
- The menu bar is always active, no matter which form you are using. Thus, you can save a file and exit from anywhere in the **Part Description Editor**.

The Part Description Editor

1 **Invoke the Part Description Editor.**

2 **Enter initial comments.**

3 **Specify other information.**

4 **Enter devices in the Device Entry Forms.**

5 **Save your file and exit.**

1 **Invoke the Part Description Editor.**

a Choose **Run Part Description Editor** from the Programs menu in IPG Test Consultant; or,

b Execute `partforms` on the BT-BASIC command line.

Each time you run the **Part Description Editor** the Current Task defaults to Initialization. The Initialization task lets you specify global information for the part description library before you begin entering information about individual devices.

c Select a Part Description Library File by selecting **Open** in the **File** menu.

You must choose a name for a new part description library file or provide the name of an existing file you wish to edit. The current path is shown in the text box below **Selection**. If **Selection** specifies a directory, the names of any files in that directory appear in the list below Files.

d Click the mouse's button on a file name to select it. Or, create a new file by typing its pathname in the box below Selection.

e When you are done, click the mouse's left button on **OK** to load the specified file and remove the Select Part Library box. To exit the selection box without taking any action, click the mouse's left button on **Cancel**.

NOTE

Double-clicking the mouse's left button on a file in the list selects it and automatically loads it into the **Part Description Editor**. This is quicker than selecting the file, then clicking on **OK**.

2 Enter initial comments.

You might want to add descriptive comments at the beginning of a part description library file. Or you may want to view the comments associated with an existing file.

a Move the mouse cursor into the text box beneath Initial Comments and click the mouse's left button to move the typing cursor into the box.

b Enter the comments.

If the initial comments are too long to fit inside the text box, use the mouse's left button to drag the scroll bar up and down to view them.

3 Specify other information.

Other information that you specify in the Initialization Form is shown in **Table 3-25** on page 3-68.

.

Table 3-25 Information specified in the initialization form

Information	Description
List of Unused (noconnect) External Pins	A list of pin numbers that are externally accessible but do not connect to anything inside the part description library device. Select this text box and enter the pin numbers that are not connected to anything. Use either a comma or a space as a delimiter (separator) between pin numbers.
Number of Internal Nodes	The total number of internal nodes that are inside the part description library. An internal node is a connection between child devices where the connection does not connect to a pin that is externally accessible. Select this text box and enter the total number of internal nodes. Then press **Return** to update the text box below Internal Nodes with the appropriate number of rows. If the internal node cannot be probed manually, click the mouse's left button on the box labeled No Manual Probe Access for that row. The box becomes highlighted to show that manual probing is not permitted for that node.

4 **Enter devices in the Device Entry Forms.**

The buttons on the Device Entry Form, and what they mean are explained in **Table 3-26**.

Table 3-26 Device entry form buttons

Button	Description
Add/Replace Device	Click the mouse's left button on this button to add the new device described in the form or to replace an existing device with the new description in the form.
Delete Device	Click the mouse's left button on this button to delete the entry for the current device.
Reset Form	Click the mouse's left button on this button to clear the current values from the form and start over. This clears the data entry fields and restores the default values where applicable (such as units of measure).

After completing the Initialization tasks, use the device entry forms to enter descriptions of the devices inside your part description library. The values you enter in these forms are merged with PDL statements and then used to create a part description library file.

Each form provides data entry fields in which you specify values for a specific kind of device—such as capacitor, resistor, or transistor. You can run the required form in either of two ways:

a Invoke the Task Menu from the menu bar and select **Device Entry** to invoke a list of forms. Then select a form from the list; or,

b If you are modifying an existing part description library, you can go directly to a specific device by clicking the mouse's left button on a device designator below List of Devices in Part Description: in any form. This automatically invokes the appropriate form and displays the description for the chosen device.

Fields with buttons by them—such as Value, Replaceable, Type, and Testable provide a shortcut that simplifies data entry. You can use the mouse to invoke a pull-down menu that contains all possible values for that option. This reduces the amount of typing required when a field has few possible values; for example, Type can be **Fixed** or **Variable**.

Below Connections are two fields, Pin 1 and Pin 2, that correspond to the two connections on a resistor. Pin 1 and Pin 2 are arbitrarily chosen designators that let you distinguish one end of this resistor from the other. The example shows that the pin denoted Pin 1 connects to external pin 1 and the pin denoted Pin 2 connects to external pin 2.

5 **Save your file and exit.**

a Select **Save** from the **File** menu to save a newly created file or to replace an existing file with the edited version; or,

b Select **Save As** from the **File** menu and specify a new name for the file. This is useful if you are modifying an existing part description library file, but want to keep the original intact.

c Unless you want to work on another file, invoke the **File** menu and select **Exit** to end your editing session and exit the Part Description Editor

Creating Custom Library Tests: Reference

This section provides instructions for creating custom library tests. These instructions include: digital in-circuit library tests, digital functional library tests, analog functional library tests, and mixed library tests.

NOTE

Digital testing is described in Chapter 1, **Digital Theory & Testing** in *Test Methods: Digital*. Analog functional testing and mixed testing are explained in Chapter 4, **Analog Functional and Mixed Testing** in *Test Methods: Analog*.

1 **Create the custom library directory**

2 **Create the library test files**

3 **Create the SAFEGUARD file**

4 **Compile the library tests and SAFEGUARD file**

Use BT-BASIC to create custom library tests and the `safeguard` file.

Store custom library tests in a custom library directory. IPG uses the library tests to generate executable tests. IPG stores the executable tests in the appropriate directories.

5 Create the custom library directory

You need to create a custom library directory for the custom library tests that you plan to write.

You can use whichever naming convention fits your needs. A common convention is to place custom library tests in a directory called `custom_lib`. You can use more than one custom library directory.

If the custom library tests are used for only one board, create the directory inside the local board directory. Whenever the entire directory is backed up or moved, the library is maintained. If the library tests are used for more than one board, create the directory one level higher than the individual board directories so that multiple board tests can use the one library directory.

Open a BT-BASIC window and type:

```
create dir "<path>/custom_lib"
```

and press **Return**.

6 Create the library test files

There are four kinds of custom library files as shown in **Table 3-27**.

Table 3-27 Custom library tests

Test	Description
Custom Digital In-Circuit Library Tests	Use digital in-circuit tests to test single digital devices with power applied to the board under test. A digital in-circuit library test consists of a single file in the custom library directory and has the same name as the device part number entered in Board Consultant (for example, cd4083). A digital setup-only test consists of the declaration section only. This provides the fixture generation software with enough information to assign test resources and fixture wiring.
Custom Digital Functional Library Tests	Use digital functional tests to test powered digital clusters with backtracing. Digital functional tests differ from digital in-circuit tests in that they usually test multiple devices together in a cluster, and allow backtracing of internal nodes, while digital in-circuit tests do not. A digital functional library test consists of a single file in the custom library directory and has the same name as the parent designator entered in the Internal Device Entry Form in Board Consultant. A digital setup-only test consists of the declaration section only. This provides the fixture generation software with enough information to assign test resources and fixture wiring.

Table 3-27 Custom library tests (continued)

Test	Description
Custom Analog Functional Library Tests	Use analog functional tests to test analog devices with power applied to the board under test. Analog functional devices are devices such as operational amplifiers (you can also test these with a digital test), voltage regulators, and oscillators. Unlike the digital library tests, the analog functional library tests need to be written specifically for the device and the topology of your board under test. IPG does not edit analog functional tests. If IPG detects a problem in the analog functional test, such as an inaccessible node or unused pins, IPG specifies the test as `nulltest` in the `testorder` file. This means that the test is not compiled for requirements in Step 3. Always examine the `ipg/summary` file after running IPG. If IPG has specified any tests as `nulltest`, you need to correct the tests and re-run IPG. An analog functional test that is specified as `comment` in the `testorder` file is compiled for requirements but is commented out in the testplan. (An exclamation point is placed in the line of code, causing the editing software to ignore the line of code.) You can edit and uncomment the test later. An analog functional setup-only test consists of `disconnect`, `clear connect`, and `connect` statements inside an analog functional test block. An analog functional test block is delimited by the `test powered analog` and `end test` statements. Unlike digital library tests, analog library tests do not need an `SAFEGUARD` file reference. IPG places analog functional tests in the `analog` directory under the local board directory. If the analog directory does not exist, IPG creates it.

Table 3-27 Custom library tests (continued)

Test	Description
Custom Mixed Library Tests	Use mixed tests for devices that are part digital and part analog, such as digital-to-analog converters. Custom mixed library tests include an analog library source (analog) and a digital library source (digital). The mixed library test directory is compiled to create `analog.o` and `digital.o` library object files. The mixed library test must consist of an analog source file called `analog` and a digital source file called `digital`. These files must be placed under the test directory in the custom library. Unlike the digital portion of mixed library tests, the analog portion must be written specifically for the device and topology of the board under test. IPG does not edit the analog portion of mixed tests. IPG treats the digital portion of mixed tests just as it does digital tests. The analog portion of a mixed setup-only test consists of `disconnect`, `clear connect`, and `connect` statements inside an analog functional test block. A mixed test block is delimited by the `test powered mixed` and `end test` statements. IPG places mixed tests in the `mixed` directory under the local board directory. If the `mixed` directory does not exist, IPG creates it.

You can write complete tests or setup-only tests. The setup-only tests include just enough information for the fixture generation software to assign testhead resources.

Copy a test from the 3070 Series 3 standard libraries to your custom library directory.

Sometimes you can find a standard library test for a device that is similar to the device you want to test, then modify a copy of it to meet your needs. This simplifies the process and helps you maintain the structure of the library tests. In addition to the standard libraries, there are analog functional test templates under `$AGILENT3070_ROOT/library/templates`. You can copy these templates to your custom library test directory and edit them to test your particular devices.

If you copy a test from the standard libraries to your custom library, be sure to name it differently from the standard library test name, or be sure the library pathnames are listed (in the Library Paths Entry Form) in an order such that IPG finds the custom version before the standard version.

NOTE

For specific information on analog functional and mixed tests see Chapter 4, **Analog Functional and Mixed Testing** in *Test Methods: Analog*; for information on digital in-circuit and digital functional tests see Chapter 1, **Digital Theory & Testing**in *Test Methods: Digital*.

7 Create the SAFEGUARD file

You need to create an SAFEGUARD file, `safeguard`, in the custom library directory if it does not exist. The `safeguard` file must reference all library tests for devices that you entered in Board Consultant with SAFEGUARD turned on (**SAFEGUARD Y**). Digital functional library tests do not need a reference in the `safeguard` file.

We recommend that you use SAFEGUARD for each digital test or digital portion of a mixed test. Digital functional tests (cluster tests), however, do not need a safeguard reference; the safeguard information for a cluster is taken from the safeguard of each individual device.

8 Compile the library tests and SAFEGUARD file

Board Consultant compiles the custom library tests and the `safeguard` file. If you are not using Board Consultant, manually compile all library tests and the `safeguard` file at this step.

To compile your tests and safeguard file, type:

```
compile "custom_lib/<device_name>"; library
compile "custom_lib/<mixed_dev_dir>"; library
compile "custom_lib/safeguard"
```

and press **Return** after each line. Note that for mixed tests, you compile the mixed test directory name.

After compilation there should be a `<device_name>.o` for each test and a `safeguard.o` under `custom_lib`. If any warnings are generated, you should evaluate them, make any needed corrections, and re-compile.

4

Generating Tests and Fixture Files

In this chapter...

Objectives

When you finish reading this chapter, you should be able to:

- Generate the initial fixture files with the first two sections of the Fixture Generation Software - Board Placement and Probe Select.
- Generate the device tests with Agilent IPG.
- Generate the test program with the Testplan Generator (TPG).
- Create custom executable tests.
- Generate test requirements files.
- Generate the final fixture files and reports with the last two sections of the Fixture Generation Software - MPA and Fixture Tooling.
- Examine the fixture files and reports.
- Generate test object files.

Prerequisites

Before you begin using this chapter, you should have:

- Examined all previously created files (such as the `board` and `board_xy` files) and provided all custom library tests (at least setup-only) before continuing. You can add custom executable tests any time before generating requirements files.

Required Tools and Materials

To accomplish the tasks in this chapter, you need:

- An Agilent 3070 Series 3 workstation

  Parameters in the `.hp3070` file help determine the performance of the software. The `.hp3070` file is in the home directory. If you want to change some of the parameters for a specific board, you can copy the complete `.hp3070` file to the board directory and edit the appropriate parameters. For example:

  ```
  copy "/users/user1/.hp3070" to
       "/users/user1/<board>/.hp3070"
  ```

  The software looks for the `.hp3070` file in the board directory first; if it does not find the file in the board directory, it looks in the home directory.

  When you run the software from Agilent IPG Test Consultant, the software looks in the board directory that you selected. When you run the software from BT-BASIC, the software looks in the board directory from which you ran BT-BASIC.

  For multiple-board fixtures, place an identical `.hp3070` file in each board directory.

Generate Initial Fixture Files

The Fixture Generation Software generates the fixture files and reports necessary to build a test fixture.

The first two sections of the Fixture Generation Software are:

- **Board Placement (BPL)** creates the fixture directory and the `fixture.o` file. Board Placement decides the placement of the board on the fixture.
- **Probe Select** determines the best type of probe and probing location for each node.

The inputs to the Fixture Generation Software are shown in **Table 4-1**. The outputs of the Fixture Generation Software are shown in **Table 4-2**.

Table 4-1 Inputs to the fixture generation software

Input	Description
`config.o`	The resources available in the testhead.
`board_xy.o`	The X-Y data.
`board.o`	The board topology information.
`fixture_defaults`	Contains custom values needed to initialize the fixture.
`custom_fix`	Contains custom fixture component files

Table 4-2 Outputs of the fixture generation software

Output	Description
`fixture.o`	A file created by Board Placement and used interactively with the other three Fixture Generation Software sections. When finished, this file contains all the fixture information. Note that only the object file is maintained in the fixture directory. The `fixture` file and its syntax are discussed in Chapter 5, **The Fixture Files** in *Data Formats*.
`fixture/summary` `fixture/details`	These reports include status information from each fixture generation section.

ADVICE

Before running Fixture Generation Software, determine if board defaults and user-modified fixture component functionality is desired for the board(s) under development. See **Create Fixture Defaults and User-modifiable Fixture Component Files** on page 3-7 in the **Creating Board Information** documentation for more information.

If you modify the fixture defaults or user-modifiable fixture components after generating the initial fixture files, you must remove the fixture directory and regenerate the fixture files.

To generate the initial fixture files:

1 **Start IPG Test Consultant.**

2 **Set the test regeneration behavior.**

3 **Set the fixture format.**

4 **Start the Develop Board Test Function.**

5 **Determine the position of the board on the fixture.**

6 **Specify fixture keepout areas.**

7 **Specify fixture wire colors.**

8 **Generate the initial fixture files.**

9 **Make changes as necessary.**

When running incrementally, you can control how IPG Test Consultant instructs Agilent IPG to regenerate tests for an existing board test. This has no effect for new board tests; IPG Test Consultant performs a complete generation for new board tests.

The Test Regeneration Behavior field of IPG Test Consultant's main menu specifies the type of regeneration you want. These types are shown in **Table 4-3**.

1 Start IPG Test Consultant.

If IPG Test Consultant is not already running, start it now.

NOTE

If you need more detailed instructions, see Chapter 1, **Agilent IPG Test Consultant** in *Test Development Tools*.

2 Set the test regeneration behavior.

Table 4-3 Test regeneration behavior types

Type	Description
Comprehensive Regeneration	IPG regenerates all tests that are affected by a change.
Comprehensive; Clear Permanent	IPG regenerates all tests that are affected by a change including tests that are marked `permanent`. The `permanent` keyword is removed from the corresponding line in the `testorder` file.
Limited Regeneration	IPG regenerates only tests which are directly affected by a change.
Limited; Clear Permanent	IPG regenerates only tests which are directly affected by a change, including tests that are marked `permanent`. The `permanent` keyword is removed from the corresponding line in the `testorder` file.

The default value for this field is determined by the `TestConsultant.DependencyCheck` parameter of the `.hp3070` file. If no default is specified in the `.hp3070` file, IPG Test Consultant uses Comprehensive Regeneration. See **IPG: Reference** on page 4-35.

Select the type of regeneration behavior in the main menu of IPG Test Consultant.

If you specify a limited mode:

- tests which are indirectly affected by a change are not regenerated by IPG and might need to be debugged.
- the `pins` and `shorts` files are not regenerated; for changes to nodes, specify a comprehensive mode or manually update the `pins` and `shorts` files.

  The type selected is indicated in the Dependencies Calculation menu, the Develop Board Test menu, and in the List of Tests to be Generated menu.

  For complete details of IPG test regeneration behavior, see the **IPG: Reference** on page 4-35.

3 Set the fixture format.

Specify whether the fixture is used to test multiple boards or panels.

- Select **Single Board/Panel per Fixture** for testing one board or panel of boards on a fixture.
- Select **Multiple Boards/Panels per Fixture** for testing multiple boards or panels of boards on a fixture.

This is NOT used for Agilent PanelTest or Dual-Well Shared Wiring.

NOTE

For information on these features, see **Chapter 1, Multiple-Board Tests & Fixtures** in *Optional Board Test Applications*.

4 Start the Develop Board Test Function.

a Select **Develop Board Test** from the Actions menu.

b Set the dependencies calculation starting point.

c Select **Begin Interactive Development**.

IPG Test Consultant displays the Develop Board Test menu. This menu lists the tasks necessary for your board test.

5 Determine the position of the board on the fixture.

You need to understand how Board Placement determines the position of the board and what you can do to influence the placement. For more information see the **Fixture Generation Software: Reference** on page 4-58.

There are four methods that you can use to determine the position of the board on the fixture:

- allow the software to automatically place the board;
- modify the automatic placement using Agilent Fixture Consultant;
- manually run Board Placement and include a placement specification in the board placement statement; or
- add a placement specification in the board_xy file before running Board Placement.

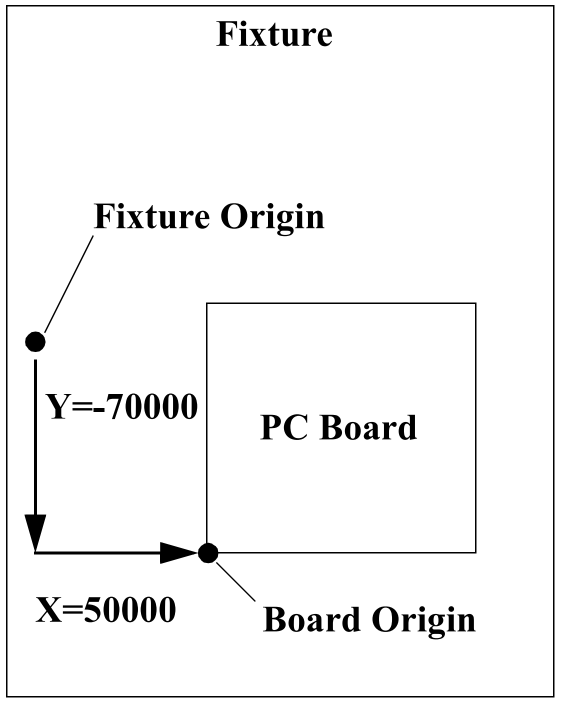
The steps to perform each of these tasks are as follows:

a Allow the software to automatically place the board. To instruct Board Placement to position the board over the center of the pin card configuration of the testhead, set the `bpl.center` parameter in the `.hp3070` file to **yes** before running Board Placement.

Board Placement includes the placement specification in the `fixture.o` file as X and Y offsets from the fixture origin to the board origin and a rotation angle.

See **Figure 4-1** for an example board placement on a bank 2 fixture. In the example, the board placement has an X offset of 50000, a Y offset of-70000 tenth-mils, and a rotation of 0.0 degrees.

Figure 4-1 Board placement for a bank 2 fixture



b Modify the automatic placement using Fixture Consultant.

If, after running Board Placement, you need to change the position of the board, use Fixture Consultant.

Fixture Consultant adds the placement specification to the `fixture.o` file as X and Y offsets from the bank 2 fixture origin to the board origin, and a rotation angle with respect to the board origin.

c Manually run Board Placement and include a placement specification in the `board placement` statement. To do this, specify the X direction offset, the Y direction offset, and the rotation. If you do not want the board rotated, specify a rotation of 0.

Then, use the `center` keyword to instruct Board Placement to position the board on the fixture so that the board is centered over the testhead resources (configuration); not necessarily centered on the test fixture. This can help minimize fixture wire lengths. If you specify only the X offset or the Y offset, and include the `center` keyword, the board is centered over the testhead resources for the direction that was not specified.

d Add a placement specification in the `board_xy` file before running Board Placement, by manually editing the `board_xy` file.

NOTE

See Chapter 2, **The board_xy File** in *Data Formats* for a description of the format and syntax of the `board_xy` file.

Use the `PLACEMENT` keyword to specify the board placement, then re-store and compile the `board_xy` file.

You must specify a complete placement; it must include X and Y offsets from the bank 2 fixture origin to the board origin, and a rotation angle. You cannot specify a partial placement in the `board_xy` file. For example:

```
PLACEMENT <X offset>, <Y offset> <rotation>;
PLACEMENT 25000, -60000 0.0;
```

Remember that the board is rotated first (and the board origin moves with the board), then the board is moved in the X and Y directions. For bank 1 fixtures, add 142500 tenth-mils to the X offset.

6 Specify fixture keepout areas.

a Use custom fixture defaults or manually add fixture keepout areas to the `fixture.o` file.

See Chapter 5, **Fixture Defaults File**, section **Keepout - (Fixture)**, in the *Data Formats* documentation for more information how to use custom fixture default file.

Use the `list object` statement to create a source listing of the object file:

```
list object "fixture/fixture.o" to
            "fixture/fixture"
```

b Load the fixture source file and add the fixture keepout areas.

A keepout area must be specified as a series of at least three points (in fixture coordinates relative to the fixture origin) that define a polygon. You can specify the side of the fixture for the keepout area—top, bottom or both. If you don't specify a side, the software assumes the bottom side. Each `keepout` keyword specifies one keepout area.

NOTE

See Chapter 5, **The Fixture Files** in *Data Formats* for a description of the format and syntax of the `fixture.o` file.

c Save and compile the `fixture` file before returning to Develop Board Test to run Probe Select.

7 Specify fixture wire colors.

a Use the `FXT.WIRECOLORS` parameter in the `.hp3070` file to specify the fixture wire colors.

The colors that you specify with this parameter are used in the fixture wiring reports. For example:

```
FXT.WIRECOLORS: "red black blue green
                 yellow aqua white purple"
```

The first color specified is used for all non-ground power wiring. The second color specified is used for all ground power wiring. Each additional color is used for all wiring of one node each, and the wire colors are alternated.

For example, if the colors listed above were specified, power nodes would use red wires, ground nodes would use black wires, the first node would use blue wires, the second node would use green wires, and so on. After the color purple was used, the sequence would start again with blue.

The color names are limited to 20 characters and are truncated to the first six characters on the fixture building reports.

We recommend that the first two colors specified are red and black respectively.

You can use the `FXT.WIRECOLORS:` parameter to provide a message for the fixture builder. For example, you can add the number of the bin where the wire is located:

```
FXT.WIRECOLORS: "red_1 blk_2 blue_3
                 grn_4"
```

You can use the `FXT.WIRECOLORS:` parameter to provide wire color names in the local language. For example:

```
FXT.WIRECOLORS: "rot schwartz blau grun
                 gelb braun"
```

8 Generate the initial fixture files.

a Select the **Generate Initial Fixture Files** task of Develop Board Test.

b Select **Do This Step** to generate the files, or **Expand This Step** to display the individual statements used to run each section of the Fixture Generation Software.

9 Make changes as necessary.

The Fixture Generation Software operates in one of two modes: new or incremental.

- In new mode, the fixture file system is created, probes and resources are initially assigned to nodes, and files and reports are generated for building the fixture.
- In incremental mode, each section of the software tries to minimize its effects on fixture component assignments (probes, nodes, wiring) and the resulting fixture files.

If the fixture has not been built, remove the existing fixture files (including `wirelist.o`) and fixture directory. This causes the software to run in new mode which optimizes resource assignments and places no unnecessary restrictions on the Fixture Generation Software; otherwise, the software assumes the incremental mode.

NOTE

If you make changes after building the test fixture, do not delete any fixture files.

The Fixture Generation Software must run in the incremental mode to minimize changes to the fixture. Fixture Tooling adds a Difference section to the

Fixture Details Report that lists the wiring changes that must be made to the fixture.

The list of inputs to the Fixture Generation Software is provided in **Table 4-1** on page 4-3, and the outputs are provided in **Table 4-2** on page 4-4.

Generate Tests with IPG

To generate tests with IPG:

1 **Select Generate Tests Using IPG.**

IPG generates the device tests, the `testorder` file, and test documentation (the `ipg/summary` and `ipg/details` files). IPG creates the `ipg` directory if it does not exist.

IPG also produces a `dependencies.o` file that lists the dependencies for each device. IPG Test Consultant uses this file, and the specified Test Regeneration Behavior, during incremental runs to determine which device tests need to be regenerated by IPG.

See the **IPG: Reference** on page 4-35 to understand the effect of test regeneration behavior.

2 **Select Modify List of Tests to be Generated.**

IPG Test Consultant displays a form that includes the three fields shown in **Table 4-4**.

Table 4-4 IPG Test Consultant form fields

Fields	Description
Regenerate	Includes a list of the tests that should be regenerated. Move any other tests that you want regenerated to this field.
Do Not Regenerate	Move tests that you do not want to regenerate this time, and you do not want marked `permanent` in the `testorder` file, to this field.
Do Not Regenerate and Mark Permanent	Includes a list of the tests that should be regenerated but are marked `permanent` in the `testorder` file. Move tests that you do not want to regenerate, and you want marked `permanent` in the `testorder` file, to this field.

After IPG Test Consultant determines which tests to regenerate, it lists those device tests in the `ipgtc.tests` file. If you are running Develop Board Test incrementally and in the single-step mode, you can change the list of tests to be regenerated by IPG or you can even skip the Generate Tests task; the time stamps of the test files and the `dependencies.o` file are updated accordingly.

a Add tests that are not listed in this form by clicking on the **Add a Test** button.

b Press the **Save** button to update and re-save the `ipgtc.tests` and `testorder` files.

3 Generate the device tests

a If you haven't done so already, select the Generate Tests Using IPG task of Develop Board Test to run IPG.

b Examine the `ipg/summary` and `ipg/details` files to see what devices have no tests or limited tests.

Make any changes that increase the test coverage of the board

c Re-run Develop Board Test.

Examine the `ipg/summary` file to see if any tests are specified as `nulltest`. This means that the test is not compiled for requirements and no resources are assigned in the fixture. You need to correct the tests and re-run IPG.

Tests that are labeled as `comment` are compiled for requirements and commented out of the testplan. You can edit the test and uncomment it later.

Generate the Test Program (Testplan)

The testplan is made up of a `testmain` that controls the testing of the board, and subroutines that include the statements to execute the device tests.

When running in the incremental mode, the testplan generator (TPG) deletes the existing testplan and generates a new testplan. You need to add all modifications to the new testplan. You can instruct TPG to leave the existing testplan and not generate a new testplan by adding the `testplan generation off` statement at the first line of the `testorder` file.

1 **Select Generate Testplan.**

   For more information, see **Testplan Generator: Reference** on page 4-49.

2 **Edit the testplan generated by TPG**

   a If necessary, enable and disable certain features by editing the Usage Flags in the `sub Set_Custom_Options` section of `testmain`.

NOTE

Chapter 6, **The Part Description Editor** in *Test Development Tools* provides more details on the Usage Flags and features.

   b If necessary, add `vacuum well` (BT-BASIC) statements to your testplan to specify which vacuum-on (`faon`, `fbon`. . .) statements turn on which vacuum wells.

   c If necessary, add `gpconnect` and `gpdisconnect` statements to the testplan to operate GP relays.

NOTE

See Chapter 3, **The Testorder File** in *Data Formats* for details of the `testorder` file and its syntax.

Create Custom Executable Tests

Create custom executable tests for any device to be tested that does not have a library test. This includes digital in-circuit, digital functional, analog functional, and mixed device tests.

For more information, see **Creating Custom Executable Tests: Reference** on page 4-52.

To write a custom test:

1 **Copy an existing test.**

Copy a test from the standard 3070 libraries to your custom executable test directory. Sometimes you can find a test for a device that is similar to the device you want to test, then modify a copy of it to meet your needs.

2 **Edit the testorder file.**

Open a BT-BASIC window and edit the `testorder` file to include all your custom executable tests.

3 **Create the test directories.**

If they do not already exist, create the following directories below the `board` directory:

- analog — for analog functional tests,
- digital — for digital in-circuit tests,
- functional — for digital functional tests,
- mixed — for mixed analog-and-digital tests.

4 **Create directories for digital functional and mixed tests.**

Create the following directories:

- functional/<test name> — for digital functional tests
- mixed/<test name> — for mixed tests

5 **Create the custom executable tests.**

The complete instructions for creating custom executable tests are provided in the **Create Custom Executable Tests** on page 4-15.

6 **Return to IPG Test Consultant to Develop Board Test.**

The Create Custom Tests task is grayed out if there is a test for each test listed in the `testorder` file; otherwise, you can select **Create Custom Tests** and it prompts you to create each missing test.

Generate Requirements (.r) Files

To generate the .r files:

1 **Select the Generate Test Requirements Files.**

The Fixture Generation Software uses the requirements files to assign testhead resources for each test and to write the fixture wiring instructions. IPG Test Consultant generates the requirements files for you.

2 **Compile mixed tests.**

Specify the test-level directory in the `compile` statement. This generates a `test.o` file beneath the test-level directory.

The Fixture Generation Software needs to know the types of testhead resources required for each test. The files created by the first compilation are requirements (.r) files. Each test has a (.r) file. For example, there is a `u101.r` file in the digital directory for the test of U101. MPA uses these requirements files during fixture generation to assign resources and create the `wirelist.o` file.

The object compilation (second compile) deletes the (.r) files and writes object (.o) files once the necessary resource have been assigned in the wirelist.

Generate Final Fixture Files and Reports

Ensure that your custom tests are complete and accurate before continuing with this task.

The remaining two sections of the Fixture Generation Software are:

- **Module Pin Assignment** (MPA) assigns module interface pins to nodes and creates `wirelist.o`.
- **Fixture Tooling** (FXT) assigns and records wiring connections and produces the files and reports needed to build the fixture.

The inputs to the Fixture Generation Software are shown in **Table 4-5**. The outputs of the Fixture Generation Software are shown in **Table 4-6**. Items to consider before generating the final fixture files and reports are presented in **Table 4-7**.

Table 4-5 Inputs to the fixture generation software

Input	Description
`config.o`	The resources available in the testhead.
`board_xy.o`	The X-Y data.
`board.o`	The board topology information.
`testorder`	A list of the requirements files.
`(.r)` files	The requirements files created by the first compilation.
custom_fix directory	Contains custom fixture component files

Table 4-6 Outputs of the fixture generation software

Output	Description
`fixture.o`	A file created by Board Placement and used interactively with the other three Fixture Generation Software sections. When finished, this file contains all the fixture information. Note that only the object file is maintained in the fixture directory. The `fixture` file and its syntax are discussed in Chapter 5, **The Fixture Files** in *Data Formats*.
`wirelist.o`	A list of the wiring from node to brc for each test. This list also specifies the type of resource for each wire. Only the object file is maintained in the local board directory. The `wirelist` file and its syntax are discussed in Chapter 4, **The Wirelist File** in *Data Formats*.
Fixture building files and reports	These are the files and reports necessary to build your test fixture. The files and reports are explained in the **Reading the Fixture Files and Reports** on page 4-93 and in the *Building Board Test Fixtures* documentation.
`fixture/summary` and `fixture/details`	These reports include status information from each of the four fixture generation sections.

Table 4-7 Considerations before generating the final fixture files and reports

Consideration	Description
Fixture Consultant	You can specify some of the attributes of your test fixture.
Fixture Generation Software Options	You can use options to influence the software.
Fixture Overflow	There are actions you can take when there are not enough resources in the testhead.
Extra Holes for Future Considerations	The Fixture Generation Software provides extra holes for future considerations.
Fixture Safety Shroud	Your test fixture might require a safety shroud.
SimPlate Fixtures	There might be blocked resources in your test fixture.

To generate the final fixture files and reports, follow these steps:

1 **Specify test fixture features.**

2 **Specify the drilled option for MPA.**

3 **Specify the options for fixture tooling.**

4 **Resolve the fixture overflow, if necessary.**

5 **Specify extra holes for future considerations.**

6 **Insert the fixture safety shroud.**

7 **Declare blocked resources.**

8 **Run the fixture generation software.**

9 **Make necessary changes.**

10 **Add remote sensing for analog tests.**

11 **Add the safety disable.**

12 **Examine the wirelist.o file**

1 **Specify test fixture features.**

a Use Fixture Consultant to specify some of the features of your test fixture.

You can specify features before generating the final fixture files; or, run the software first, then make any necessary changes. Features that you can specify are shown in **Table 4-8**.

b Use the graphical wiring diagram of Fixture Consultant to verify areas of concern before building the fixture.

Table 4-8 Features of your test fixture

Feature	Description
Board Placement	Must be done before generating final files and reports.
Probe Type	The type of probe to be used on a particular node.
Probe Location	The location to be probed on a particular node.
Extra Resource Attributes	Blocked, drilled, or drilled and receptacles inserted.
Wiring Paths	Control wiring paths and add brc's to nodes.
Fixture Electronics	Add wiring for fixture electronics.

2 **Specify the drilled option for MPA.**

Drilled instructs MPA to use only resources that have partially drilled holes or holes that are completely drilled with receptacles inserted.

3 **Specify the options for fixture tooling.**

These options are described in **Table 4-9**.

Table 4-9 Options for fixture tooling

Option	Description
cartesian	Instructs Fixture Tooling to generate X-Y data on a vertical Y axis and a horizontal X axis. If cartesian is not specified, the X-Y data is on a vertical X axis and horizontal Y axis.
terse	Instructs Fixture Tooling to generate fixture reports with no headings or white space.
minimize	Instructs Fixture Tooling to assign only partially drilled holes for extra resources instead of completely drilled holes with receptacles inserted.

4 Resolve the fixture overflow, if necessary.

A fixture overflow occurs when not enough resources are available in the testhead. The fixture generation process continues to determine and report the additional resources needed, but MPA creates a `wirelist` file only, and not a wirelist.o file. Fixture Tooling does not run. To resolve a fixture overflow:

a Limit the resource requirements for the board test by deleting tests, lowering the accuracy of tests (increasing the tolerance multiplier), turning off Remote Sensing, or limiting the amount of upstream disabling and conditioning.

b Override blocked resources and plan to use offset personality pins.

c Increase resources in the testhead by adding more pin cards to the testhead.

The maximum number of interface pins in each module varies depending on the type of cards installed.

5 Specify extra holes for future considerations.

Because it is very difficult to add holes to an existing fixture, Fixture Tooling assigns some extra holes that the Fixture Generation Software can use when running in incremental mode. Partially drilled holes are listed in a separate section of the drill file. Extra holes with receptacles are listed as EXTRA in the fixturing reports.

The following extra resources have holes drilled and receptacles inserted:

- GP relays,
- clock receiver,
- user clock,
- up to eight personality pins per HybridPlus Card (personality pins do not use receptacles), and
- all AccessPlus Card resources.

The following extra resources get partially drilled holes:

- safety disable,
- fixture id (autofile),
- event trigger, and
- remaining unused power supplies in modules that have one or more power supplies used in the test.

6 Insert the fixture safety shroud.

Be sure that the test and fixture do not present any safety hazards to the operator. The shroud protects the operator if any of the following are true:

- Voltage values equal to, or exceeding, the following levels (with respect to chassis ground) are present on the board under test:
  - 60 volts dc
  - 42 volts peak
  - 30 volts RMS
- The test and fixture apply power to electrical or electronic devices that could explode when defective or mis-loaded on the board under test.

7 Declare blocked resources.

Due to mechanical considerations, there are blocked resources in row 23.5 on Bank 2 for Agilent SimPlate test fixtures. Declare these resources as blocked before generating the final fixture files.

The blocked resources are:

```
223153 223154 223155 223156 223157 223158
223159 223160 223161 223162 223163 223164
223165 223166
```

a Use Fixture Consultant to declare these blocked resources; or,

b List and edit the `fixture.o` file.

```
list object "fixture.o" to "fixture"
```

c Manually add the brcs to the PINS (OTHER) section, or merge the `$AGILENT3070_ROOT/contrib/lib/SimplateBlockedPins` file.

d Re-save and compile the `fixture` file.

NOTE

With Agilent 3070 software revision 3070 04.00pa, an environment variable was created so that files can be easily transferred between UNIX® and MS Windows® controllers, which have different file systems. The environment variable, `$AGILENT3070_ROOT`, replaces the upper path names on both systems. For example, the `$AGILENT3070_ROOT` factory default value is /var/hp3070. In this document, only path names using the environment variable are used. If you must use actual path names, refer to older versions of the documentation. Please see **The Root Directory Environment Variable** in *Administering Agilent 3070 UNIX Systems* for further information.

8 **Run the fixture generation software.**

a Select the **Generate Final Fixture Files and Reports** task of Develop Board Test

b Select **Do This Step** to generate the files, or **Expand This Step** to display the individual statements used to run each section of the Fixture Generation Software.

9 **Make necessary changes.**

Sometimes you need to make changes to the board or fixture information. If you make changes after generating the fixture files and reports, but before building your test fixture, delete the entire fixture directory and the wirelist.o file (also the `wirelist` file if it exists) before re-running Develop Board Test. This makes your fixture reports and files original and places no unnecessary restrictions on the Fixture Generation Software. Be aware that if you had edited the `fixture.o` file, such as adding fixture keepout areas, you need to make the same edits to the new `fixture.o` file.

If you make changes after building the test fixture, do not delete any fixture files. The Fixture Generation Software must run in the incremental mode to minimize changes to an existing fixture. Fixture Tooling adds a Difference section to the Fixture Details Report that lists the wiring changes that must be made to the fixture.

10 Add remote sensing for analog tests.

If necessary, manually add remote sensing for accurate analog in-circuit tests.

a Use Fixture Consultant to add a sense wire and probe.

b Edit the device test to add the bus connections and sense options.

c Run Develop Board Test in IPG Test Consultant to compile and update all necessary files.

d Perform the necessary changes to the fixture.

11 Add the safety disable.

The 3070 Series II and 3 board test systems provide a safety disable to interrupt power to the board under test if the safety shroud is opened.

The safety disable is on the Control Card (pins 43 and 44). The test fixture includes a jumper across pins 43 and 44 of the control card for each module (two jumpers in a single-bank fixture, and four jumpers in a full size fixture). The testhead configuration file determines which module is used for the safety disable; the module who's description includes the `probe` statement is used for the safety disable. To be sure the safety disable is wired correctly for any testhead, disconnect all jumpers across pins 43 and 44, wire all pins 43 to one side of the safety shroud interlock switches, and wire all pins 44 to the other side of the safety shroud interlock switches.

The safety feature is turned on at the first occurrence of a `powered` or `auxconnect` statement in the testplan. The safety feature is turned off at the last occurrence of an `unpowered` or `auxdisconnect` statement.

The safety shroud can be open or closed anytime the safety feature is not turned on.

Similarly, if the shroud is open when the testplan starts, the test stops at the first occurrence of a `powered` or `auxconnect` statement and issue a warning message.

If the shroud is opened while the safety feature is turned on, the following occurs:

- the test stops
- all DUT power supply relays, all functional port relays, and all external port relays (on the AccessPlus and Serial Test Card) are opened to disconnect potential high voltage sources to the DUT
- vacuum is turned off
- a warning message is issued

12 Examine the wirelist.o file

a Use MPA to create the `wirelist.o` file.

b Use the `list object` statement to create a source listing. For example:

```
list object "wirelist.o" to "wirelist"
```

c Load and examine the `wirelist` file. Editing the `wirelist` file is not recommended. MPA, in the incremental mode, updates the `wirelist` source file, if one exists, in the board directory.

Examine the Fixture Files and Reports

1 **Verify the fixture reports and files.**

The Plot Generator creates files, which you can copy over a plotter, from information in the `fixture/fixture.o` file. IPG Test Consultant automatically runs the Plot Generator when you select the **Generate Fixture Files and Reports** task from the Develop Board Test menu. You can run the Plot Generator manually with the `generate plot` statement. The Plot Generatorcreates the plot files described in Table 4-10 and places them in the `fixture` directory.

NOTE

See Chapter 5, The Plot Generator in *Test Development Tools* for details of the Plot Generator Web Service.

Table 4-10 Plot files created by the plot generator

File	Description
probes	Probe and personality pin locations. This file is called `probes.p`. If top probes are allowed, another file (`probestop.p`) is created for the probe locations (not personality pins) in the top probe plate.
inserts	Type and location of inserts. This file is called `inserts.p`. If top probes are allowed, another file (`insertstop.p`) is created for the inserts in the top probe plate.
wires	Critical and functional wires. This file is called `wires.p`. If top probes are allowed, another file (`wirestop.p`) is created for the critical and functional wires in the top probe plate.
alignment	Personality pin locations with respect to the alignment plate. This file is called `alignment.p`.

2 **Fix errors.**

a Modify the appropriate file (usually `board_xy`)

b Re-run **Develop Board Test**.

If your fixture has not been built, you should delete your `fixture` directory and `wirelist` files before running Develop Board Test. Re-create any manual changes such as board placement or fixture keepout areas.

CAUTION

If your fixture has already been built, do not delete your fixture directory or wirelist files before running Develop Board Test. The Fixture Generation Software appends a Differences report to `fixture/details` to list modifications that need to be made to the fixture.

Generate Object (.o) Files

1 **Select Generate Test Object Files; or**

2 **Manually compile the tests.**

  a Compile all the tests (`pins` file, `shorts` file, device tests, and `backtrace` files) with the `compile` statement.

  b Compile mixed tests. Specify the test-level directory in the `compile` statement. This generates a `test.o` file beneath the test-level directory. You do not need to compile the board `safeguard` file as you did in the first compile.

Each test now has an object (.o) file, and the requirements (.r) files are deleted. If you have made a change and are running incrementally, any test that requires resources different from the original test might retain the (.r) file and no (.o) file is generated. In this case, run the Fixture Generation Software again. Fixture Tooling appends a Difference report to the `fixture/details` file documenting the changes necessary to the test fixture.

3 **Re-calculate dependencies.**

Select the **Re-Calculate Dependencies** task to calculate dependencies at the end of the Develop Board Test function as a final check to be sure that all files were completed in the proper order.

4 **Generate a new testability report.**

The report is complete because all files are now available. Return to Agilent Board Consultant and generate the report.

Summary

The Develop Board Test menu of IPG Test Consultant has been used to run IPG to generate the device tests and to run TPG to generate the test program testplan.

Custom executable tests have been created and listed in the `testorder` file.

The Develop Board Test menu of IPG Test Consultant has been used to generate the requirements (.r) files, and to run the Fixture Generation Software to generate the files and reports necessary to build the test fixture.

When all files were complete and accurate, the object (.o) files replaced the requirements (.r) files.

The additional directories created (under the board directory) are shown in **Table 4-11**.

The additional files created elsewhere are shown in **Table 4-12** on page 4-30.

The requirements files are shown in **Table 4-13** on page 4-34.

Table 4-11 Directories created under the `board` directory

Directory	Description
`analog`	Contains analog in-circuit and analog functional tests.
`digital`	Contains digital in-circuit tests.
`functional`	Contains digital functional tests.
`functional/<device test>`	Contains the digital functional test files. There is one directory for each digital functional test.
`mixed`	Contains mixed test directories.
`mixed/<device test>`	Contains the mixed test files. There is one directory for each mixed test.
`ipg`	Contains the results of the latest run of IPG. The `testmain` file is an optional file in this directory.
`fixture`	Contains fixture files and reports. The Fixture Generation Software creates this directory.

Table 4-12 Additional files

File	Description
`<board>/ipgtc.tests`	List of tests to be regenerated by IPG in the incremental mode. This file is unlinked after IPG has run.
`<board>/testorder`	List of the device tests. The `testorder` file is created by IPG, and is an input to Testplan Generator.
`<board>/testplan`	Program that tests your board. The testplan is an output of the Testplan Generator.
`<board>/summary`	The summary file from IPG Test Consultant.
`<board>/safeguard.o`	Compiled object file of the board-level `safeguard` file `board/wirelist.o` — Object file that contains a list of the wiring from node to brc for each test. The `wirelist.o` file is an output of MPA.
`<board>/handler_params`	Physical information about the board under test. The Agilent Express Fixturing System uses this file when testing boards. If you are not using an Agilent Express Cassette fixture, this file does not exist.
`ipg/details` and `ipg/summary`	IPG information about the tests it generated.
`ipg/dependencies.o`	List of the dependencies for each device test. This is a binary file you cannot read. It is used by IPG Test Consultant for incremental runs of IPG. This file is read and updated each time IPG is run.
`ipg/testmain`	The Testplan Generator merges this file into the testplan. It controls the testing of your board. If there is no `ipg/testmain`, TPG uses `$AGILENT3070_ROOT/standard/testmain`.

Table 4-12 Additional files (continued)

File	Description
`ipg/raw`	The information to be formatted into the `summary` and `details` files.
`fixture/fixture.o`	Object file that contains the fixture information.
`fixture/details` and `fixture/summary`	Fixture generation status information.
fixture files and reports	Files and reports necessary for building your test fixture. The fixture files and reports include: • `fixture.o` • `fixture/summary` • `fixture/details` • `wires` • `inserts` • `drill, drilltop` • `drillsup, drillgdp` • `trace` These files are in the `fixture` directory. Another file, `handler_params`, is stored under the `board` directory.
`fixture/probes.p`	Plot file of the probes information from the `fixture.o` file.
`fixture/probestop.p`	Plot file of the probes information for the top plate of the cassette fixtures. This information is from the `fixture.o` file.
`fixture/wires.p`	Plot file of the wires information from the `fixture.o` file.
`fixture/wirestop.p`	Plot file of the wires information for the top plate of the cassette fixtures. This information is from the `fixture.o` file.

Table 4-12 Additional files (continued)

File	Description
`<board>/pins`	List of nodes. It is written by IPG and used for CHEK-POINT.
`<board>/pins.o`	Compiled object version of the `pins` source file.
`<board>/shorts`	List of nodes written by IPG and used for shorts and opens testing.
`<board>/shorts.o`	Compiled object version of the `shorts` source file.
`analog/.discharge`	Capacitor discharge test written by IPG.
`analog/.discharge.o`	Compiled object version of the `.discharge` source file.
`analog/<device test>`	Analog in-circuit test file. There is one file for each analog in-circuit test. The analog tests are written by IPG.
`analog/<device test>.o`	Compiled object version of each analog in-circuit test source file.
`analog/<analog_func_test>`	Source file for each analog functional test.
`analog/<analog_func_test>.o`	Compiled object version of each analog functional test source file.
`digital/<device test>`	Digital in-circuit test file. There is one file for each digital in-circuit test.
`digital/<device test>.o`	Compiled object version of each digital in-circuit test source file.
`functional/<device test>/test`	Digital functional test source file.
`functional/<device test>/test.o`	Compiled object of each digital functional test source.
`functional/<device test>/backtrace`	Backtrace file for the digital functional test.

Table 4-12 Additional files (continued)

File	Description
`functional/<device test>/backtrace.o`	Compiled object version of each backtrace source file.
`functional/<device test>/states`	States file for the digital functional test.
`functional/<device test>/states.o`	Compiled object version of the `states` source file.
`mixed/<device>/analog`	Source file for the analog portion of a mixed test. There is one `analog` file for each mixed test directory and it must be named `analog`.
`mixed/<device>/digital`	Source file for the digital portion of a mixed test. There is one `digital` file for each mixed test directory and it must be named `digital`.
`mixed/<device>/test.o`	Compiled object file for the mixed test. There is one `test.o` file for each mixed test.

Table 4-13 Requirements files

File	Description
`<board>/pins.r`	Requirements file for CHEK-POINT.
`<board>/shorts.r`	Requirements file for the shorts and opens testing.
`analog/.discharge.r`	Requirements file for the discharge test.
`analog/<device_name>.r`	Requirements file for each analog in-circuit test.
`digital/<device_name>.r`	Requirements file for each digital in-circuit test.
`functional/<device_name>/test.r`	Requirements file for each digital functional test.
`functional/<device_name>/backtrace.r`	Backtrace requirements file for each digital functional test.
`analog/<analog_func_test>.r`	Requirements file for each analog functional test.
`mixed/<device_name>/analog.r`	Requirements file for the analog portion of each mixed test.
`mixed/<device_name>/digital.r`	Requirements file for the digital portion of each mixed test.

IPG: Reference

IPG generates device tests and test files to be run by your testplan.

IPG is run by the Develop Board Test function of IPG Test Consultant. You can run this program manually with the BT-BASIC statements `ipg`, `ipg on`, and `ipg from`.

IPG is the integrated program generator that automatically creates files for shorts and opens testing and CHEK-POINT. It also generates analog in-circuit, digital in-circuit, and digital functional tests. IPG writes the `testorder` file to be used by IPG Test Consultant, MPA, and the Testplan Generator (TPG). IPG also uses the `testorder` file when run in the incremental mode.

The inputs to IPG are described in **Table 4-14**.

Table 4-14 IPG inputs

Input	Description
`board.o`	The board description object file. This file contains information about the devices on the board and their connectivity. IPG uses device values and tolerances, and the topology of the circuit to write analog in-circuit tests. To write digital in-circuit and digital functional tests, IPG uses the device type, the topology of the circuit, and the pathnames to the library tests. IPG also gets test and fixture option information from `board.o`. This includes: tolerance multiplier, remote sensing, capacitance compensation, and digital family reference levels. See Chapter 1, **The Board File** in *Data Formats* for an explanation of the `board` file and its syntax.
`board_xy.o`	The X-Y information object file. This file contains information about the X-Y location of device.pins, tooling pin holes, and board outline coordinates. IPG finds, in this file, nodes that cannot or should not be probed. It must write the device tests to accommodate nodes that are not probed.

Table 4-14 IPG inputs (continued)

Input	Description
library tests	The standard libraries supplied with the 3070 Series II and custom libraries that you have added. The libraries can be complete tests or setup only tests. This includes digital in-circuit and digital functional tests, analog functional, and mixed tests. IPG modifies the library test (except analog functional and the analog portion of mixed tests) according to the circuit topology information to make an executable test. All setup-only tests are labeled `comment` in the `testorder` file and are listed in the `ipg/summary` file. The Fixture Generation Software uses the information in the setup-only test to reserve resources for that test.
`testorder`	IPG uses this file during incremental runs to determine which tests should not be regenerated. IPG does not regenerate tests that are labeled `permanent` in the `testorder` file.
`wirelist`	IPG uses this file during incremental runs to determine existing analog tests. IPG tries to avoid wiring changes on existing analog tests.

Running IPG

The Develop Board Test function of IPG Test Consultant runs IPG automatically for you.

IPG in New Mode

A brief description of the tasks completed by IPG running in New mode is given in **Table 4-15**.

IPG in Incremental Mode

In the incremental mode, IPG Test Consultant uses the `ipg/dependencies.o` file (created previously by IPG) and the specified Test Regeneration Behavior to determine which tests are to be regenerated by IPG. The Test Regeneration Behavior, specified in the main form of IPG Test Consultant, is described in **Table 4-16** on page 4-39.

Table 4-15 Tasks completed by IPG

Task	Description
Write pre-shorts tests	Tests to be executed before shorts and opens testing, such as potentiometer tests.
Create `shorts` file	For shorts and opens testing.
Create `pins` file	A subset of the `shorts` file that is used for **CHEK-POINT**.
Create the discharge test	Determines capacitors that might hold a charge and writes a test block to discharge them. This test is stored in the `analog` directory.
Write analog in-circuit tests	Creates the `analog` directory and stores the tests in this directory.
Write digital in-circuit and digital functional tests	Creates the `digital` and `functional` directories. IPG stores digital in-circuit tests in the `digital` directory, and digital functional tests in the `functional` directory. Digital functional tests have two files at this time, the `test` file and the `backtrace` file. During test completion and debug, you add a `states` file. The `states` file can be learned at test completion time.
Manage the analog functional and mixed tests	IPG does not modify analog tests, or the analog portion of mixed tests, for topology. If IPG finds a problem with an analog library, such as an inaccessible node, it flags the `testorder` file entry for that test as `nulltest`. This means that resources are not assigned for that test. You need to correct the library test and re-run IPG. You can use the `ipg on` statement to run IPG on individual devices. IPG treats the digital portion of mixed tests just as it does digital tests. If IPG modifies the digital portion of the mixed test, IPG flags the mixed test as `comment` in the `testorder` file.

Table 4-15 Tasks completed by IPG (continued)

Task	Description
Write the `summary` and `details` files	These files contain information about the tests that IPG wrote. These are the `ipg/summary` and `ipg/details`; do not confuse them with the `fixture/summary` and `fixture/details`. Always view the `summary` file to determine which tests were marked `nulltest` in the `testorder` file. Correct the necessary library tests and re-run IPG to ensure that the Fixture Generation Software assigns resources for all appropriate tests.
Create the `testorder` file	This is a list of the tests and the test type, in the order that they are executed. This file is an input to the Testplan Generator (TPG), IPG Test Consultant, and MPA.
Create the `dependencies.o` file	A list of dependencies for each device test. This list is used by IPG Test Consultant to determine which device tests need to be regenerated because of changes to the board description.

Table 4-16 Test regeneration behavior

Behavior	Description
Comprehensive Regeneration	IPG regenerates all tests that are affected by a change.
Comprehensive; Clear Permanent	IPG regenerates all tests that are affected by a change including tests that are marked `permanent`. The `permanent` keyword is removed from the corresponding line in the `testorder` file.
Limited Regeneration	IPG regenerates only tests which are directly affected by a change.
Limited; Clear Permanent	IPG regenerates only tests which are directly affected by a change, including tests that are marked `permanent`. The `permanent` keyword is removed from the corresponding line in the `testorder` file.

The default value for this field is determined by the TestConsultant.DependencyCheck: parameter of the `.hp3070` file. The values for this parameter are:

- `comprehensive` (Comprehensive Regeneration)
- `comprehensive_clear` (Comprehensive; Clear Permanent)
- `limited` (Limited Regeneration)
- `limited_clear` (Limited; Clear Permanent)

If no default is specified in the `.hp3070` file, IPG Test Consultant uses Comprehensive Regeneration.

In a limited test regeneration mode IPG Test Consultant instructs IPG to regenerate:

- tests that are directly affected by a change in device type, value, or connectivity;
- tests that use a changed library;
- boundary-scan device connect tests that use a changed BSDL library and the corresponding interconnect test; and
- `.discharge`, Agilent TestJet, and Agilent Polarity Check tests if any related device tests are also regenerated.

In a limited mode, IPG does not regenerate the `pins` and `shorts` files. If there are changes to nodes, we recommend that you use a comprehensive mode or manually update the `pins` and `shorts` files.

When IPG Test Consultant has determined which tests are to be regenerated, it lists those tests in a file called `ipgtc.tests`.

IPG Test Consultant provides a form that you can use to change the list of test in the `ipgtc.test` file. To display this form select **Modify List Of Tests To Be Regenerated** from the action list of the Create Tests Using IPG task.

Use the mouse to move the device tests to the appropriate fields and press the **Save** button to update and re-save the `ipgtc.tests` and `testorder` files.

IPG Test Consultant runs IPG on the tests listed in the `ipgtc.tests` file.

IPG Outputs

After running IPG your board directory includes the files and directories in **Table 4-17**.

Table 4-17 IPG output files and directories

File/Directory	Description
`pins` file	This is a subset of the `shorts` file and is used for CHEK-POINT.
`analog` directory	This contains discharge, pre-shorts, analog in-circuit, and analog functional tests. If you create executable analog functional tests, you have created this directory manually before running IPG.
`digital` directory	This contains digital in-circuit tests
`functional` directory	This directory contains digital functional tests. This includes the `test` and `backtrace` files only. The `states` file is provided or learned later.
`mixed` directory	This directory contains a sub directory for each mixed test. If you create executable mixed tests, you have created this directory manually before running IPG. If your board test contains no mixed tests, this directory is not created.

Table 4-17 IPG output files and directories (continued)

File/Directory	Description
`testorder` file	This file lists all the device tests. This file is used by the Testplan Generator. If you create custom executable tests, you have created this file before running IPG. In this case, IPG reads the `testorder` file and adds the necessary lines to it.
`ipg/summary` file	This files presents general information about the tests generated, and the program generator options used by IPG.
`ipg/details` file	This file gives specific information on each analog, digital, and functional test. You can use this information to debug and optimize the device tests.
`ipg/dependencies.o`	This file lists the dependencies for each device test. This file is used by IPG Test Consultant during incremental runs to determine which device tests need to be regenerated by IPG.

The IPG Summary Report

The IPG `summary` report, `ipg/summary`, provides a brief summary of the results of device test generation. The information in the `summary` report is formatted from the `ipg/raw` file. The `summary` report includes:

- A list of the digital devices on the board. This list includes the number of:
  - total units
  - complete tests
  - units retained
  - full tests
  - total vectors
  - partial tests
  - vectors retained
  - empty tests
  - bus enabled tests
  - setup tests
  - non-digital tests
  - analog null tests
- A list of the analog devices on the board. This list includes the number of:
  - limited analog
  - commented analog
- A list of digital tests that had an enabled bus problem.

- A list of digital tests that had no vectors removed.
- A list of digital tests that exercise all pins, but have had some vectors removed.
- A list of digital tests that had some vectors removed.
- A list of digital tests that had all vectors removed.
- A list of digital tests that are setup-only tests.
- A list of analog or mixed tests that had a pin missing from the `board.o` file, or a pin that is inaccessible. Either correct these tests or specify them as **NOT TESTED** in the `board.o` file.
- A list of digital tests that had all digital pins removed.
- A list of limited analog tests.
- A list of commented analog tests.

The IPG Details Report

IPG places detailed information about the device tests in two files:

- The `ipg/details` file — includes information about backtrace tests.
- The `ipg/raw` file — includes information about the following types of tests: shorts and opens, CHEK-POINT (`pins`), capacitor discharge, digital and analog.

You can load or view the `ipg/details` file with a BT-BASIC window or IPG Test Consultant. You can execute the format program with the `fmt` statement in a BT-BASIC window to format the information in the `ipg/raw` file.

a To access the information about one test in the `ipg/raw` file, execute the following statement from a BT-BASIC window:

```
execute "fmt -t <test name> ipg/raw";wait
```

b To save the information about one test from the `ipg/raw` file in another file, execute the following statement from a BT-BASIC window:

```
execute "fmt -t <test name> ipg/raw
        <file name>"
```

You can then load or print that file to view the information.

c To format the information about all tests in the `ipg/raw` file and place the information in a file, execute the following statement from a BT-BASIC window:

```
execute "fmt ipg/raw <file name>"
```

d To generate a summary of the information in the `ipg/raw` file, execute the following statement from a BT-BASIC window:

```
execute "fmt -s ipg/raw"; wait
```

The information in the `ipg/raw` files depends on the experience level under which IPG was run; the information might contain only a subset of the information described in this section.

NOTE

The `ipg/raw` file can contain the information given in **Table 4-18**.

Table 4-18 Information in the `IPG/RAW` file

Information	Description
Shorts Test Details	Lists the opens-threshold and all nodes that were found to be shorted including the impedance between each pair of shorted nodes. The impedance for each open node is listed. Any change in settling delay or threshold is noted. All nodes that are not accessible are also noted.
CHEK-POINT (pins) Test Details	Lists each node that is isolated (open to CHEK-POINT), or not accessible.
Analog In-Circuit Test Details	Contains a section for each analog in-circuit test. This section is described in detail in Chapter 5, **Debugging Analog Tests** *Test Methods: Analog*.
Digital Test Details	Lists, for each digital device test, which pins are not tested, which pins are under tested, which upstream pins are disabled or enabled, if a bus could not be disabled, and if pull-up resistors are required to test the device.
Warnings	General comments, such as the effect of inaccessible nodes on shorts testing and CHEK-POINT.

How IPG Generates Tests

This section explains how IPG generates tests. You do not need to know the information in this section to be able to develop and debug your board test. This section is provided only as reference if you should want to know these details.

Analog In-Circuit Tests

Based upon the board description, IPG determines which analog components are to be in-circuit tested and how to connect them to the measurement operational amplifier (MOA) circuit (on the ASRU Card). It generates the component test statements, and the relay manipulation statements necessary to connect the appropriate buses to the component under test.

Value and tolerance data is included in the description of each component. Accordingly, IPG selects a test method that allows the component to be tested within the implied accuracy constraints. Also, IPG generates the in-circuit tests under the guidelines of the device and global options.

Within these parameters, IPG generates the statements necessary to perform the in-circuit tests. It also determines which measurement options must be included in each test statement to ensure accuracy.

Finally, IPG assembles all the necessary relay manipulation and test statements into test blocks for execution during board test.

IPG uses three methods to generate analog in-circuit tests. These are:

- guarding,
- test selection, and
- modeling.

Guarding is used to break parallel paths around the component under test. The component can then be effectively analyzed by IPG.

Test selection chooses which options to use in the test measurement.

Modeling simulates the board under test and the test hardware as configured for the measurement.

In effect, IPG models the in-circuit test capability of the 3070 Series II and 3 hardware for a particular group of components. Based on this model, IPG can generate those test statements and measurement options that most reliably test a component to the required accuracy. The primary goal is to pass all good parts.

To model effectively, IPG considers factors such as: component and fixture characteristics, system noise and error, and transient analysis. To maintain test reliability and to achieve the goal to pass all good parts, IPG models parameters for approximately ten different variations of the device under test, and 3070 Series II and 3 test systems. The resulting variations are used to set the expected measurement tolerances.

Test Option Effects:

The device and global options define certain fixture/component characteristics that enable the system to adapt to different test requirements. In particular, the device and global options are used by measurement analysis to produce measurement options. The measurement options are then included in the test statements to obtain the desired test results. By selecting the proper device and global options, you can direct IPG to generate tests that ensure optimum accuracy for each test. That is, you can define the device and global options so the proper measurement options (such as enhancement, extra digit, and remote sensing) are selected for the most accurate measurement of the device under test.

Error Analysis:

For a given hardware/component configuration, IPG uses an AC and DC circuit analysis to calculate the system error of the measurement set up. Factors in this calculation include:

- Lead resistance and inductance.
- System capacitance.
- Model of testhead hardware and Auto-adjust.
- Multiple guard leads.
- Entire connected sub circuit on the board under test.

Noise Considerations:

IPG's measurement analysis includes certain noise considerations. The noise spectral density of the applied source and the MOA are evaluated so that peak-to-peak noise can be estimated.

Transient Analysis:

IPG performs transient analysis for each resistive device, potentiometers, FETs, and resistors under test. This analysis includes the effects of:

- all resistive devices and capacitors between the I bus and the S or G bus;
- the feedback resistance;
- system capacitance;
- capacitors between the S bus to the G bus; and
- the response of the MOA to wide and narrow bands.

IPG uses the results to determine wait intervals and adjust test limits due to settling uncertainty.

Wait time selection is based on the time for the measured reading to settle to a calculated window—10% of the tolerance margin times the minimum DUT tolerance. A wait greater than or equal to one second causes the test to be commented out (designated to be ignored) by IPG.

Parallel Networks:

IPG writes a test for all parallel devices. One or more of the tests are probably commented out because of excessive test limits. The failure message of any test lists all devices in parallel with it.

Test Classification:

After analyzing various test characteristics, IPG might classify a test as acceptable. If so, then the test is set up as specified. However, if a test is adversely affected by factors such as guarding error, limited test specifications, or tight tolerances it might be classified differently. In these cases, a test might be considered limited or questionable, or a component might be considered untestable. See the following formula.

An acceptable test meets this condition:

$$\mathbf{TestQualityis}\frac{6 \times \mathbf{Sigma}}{|\mathbf{X_{max}} - \mathbf{X_{min}}|} \leq \mathbf{TM}$$

Where Sigma is the standard deviation of the simulation with the device value at nominal. X_{min} is the simulation with the device value at minimum. X_{max} is the simulation with the device value at maximum. TM is the tolerance multiplier.

IPG adjusts the tolerances to minimize the possibility of rejecting marginally good devices.

A test is limited if it meets or exceeds the above condition. In this case, the best test tried is specified and a comment is placed in the testplan before the device test statement.

Occasionally, a component is considered untestable and IPG still generates a test for it; however, the test is commented out. IPG adds a comment to the analog source file to indicate why the test was commented out.

Digital In-Circuit and Digital Functional Tests

IPG uses the digital device libraries to produce the associated executable digital tests. This consists of two major functions: topology conflict resolution and device disable. As IPG encounters and resolves topology conflicts and disable situations, it prints comments to help you understand what it did, or what you need to do. The following paragraphs explain topology conflicts, device disables, IPG generated comments, and test developer generated comments.

Topology Conflicts:

A topology conflict exists if a device cannot be overdriven with its standard library program because of the circuitry around the device. Here are examples of pins that cannot be driven by the standard library test.

- If pins of the device are tied high (to VCC), they cannot be driven low.
- If pins are tied low (to ground), they cannot be driven high by the test.
- If pins are unconnected and not probed, they cannot be driven at all.
- If pins are tied to other pins on the same device, they cannot be driven to opposite states at the same time.

Pins are considered tied if they connect to the node directly, or through a jumper/fuse/switch, or through any resistance less than the short threshold value. To resolve this type of conflict, IPG comments out the necessary vectors or units in the executable test. Only vectors are commented out of combinatorial tests; the entire unit is commented out of a sequential test.

Preconditioning (Disabling and Conditioning):

The test for a device can be effected by the surrounding devices in the circuit. IPG uses information from the `board` file and from the device library tests to generate tests that minimize the effects of the surrounding devices. When IPG determines that it cannot test the device by overdriving upstream devices, it uses preconditioning.

Preconditioning sets a pin or device to some specific state, and holds it there, so as to minimize the adverse effect the pin or device has on the device under test. Preconditioning applies states to one or more inputs of the device whose outputs are to be preconditioned; it uses either of two methods: disabling the output pins if possible, or conditioning the output pins if disabling is not practical.

- **Disabling** turns outputs (including bidirectional pins) off by setting them to a high-impedance (**Z**) state. Disabled pins have no effect on the device under test. Normally off, upstream disabling can be turned on or off, for all devices on the board or for individual devices, in Board Consultant. IPG always disables busable outputs or bidirectional pins connected to the outputs or bidirectional pins of the device under test if the information is provided and the circuit allows. This feature (busable disabling) is always on and cannot be turned off, regardless of the setting of other parameters in Board Consultant.
- **Conditioning** sets outputs (including bidirectional pins) to states where they have a known effect on the device under test. Normally off, conditioning can be turned on or off for individual devices in Board Consultant. Turning conditioning on also turns **disabling** on, even if disabling is currently turned off. IPG always tries to disable a pin or device first. If it is unsuccessful, it tries to condition the pin or device.

If IPG can not eliminate the effects of surrounding devices on part of the device under test, it comments out that part of the device test. If IPG can not eliminate the effects of surrounding devices for the entire device under test, it comments the device test in the `testorder` file. The test is not executed. IPG adds comments to the device test and the `ipg/summary` file to tell you which devices and pins need to be disabled. You can use this information to add disable statements, or modify the test vectors or units so that the test, or at least part of it, can be executed safely.

IPG Generated Comments:

When IPG is running, it adds (if necessary) several types of comments to the VCL test program to help the test developer. For example, it indicates which digital units and vectors were removed from a test program, and why. It indicates which devices must be manually disabled, and it describes topology conflicts encountered in generating a test.

The comments that IPG generates are:

- The **disable** comment indicates which devices the test developer must disable because IPG could not. The disable comments appear in the declaration section of the test. Here are some examples:

  ```
  ! IPG: user must disable U1.5
  ! IPG: user must disable U6.13
  ```

  IPG also indicates which devices it has disabled in the test. Here is an example:

  ```
  ! IPG: disabled device outputs
       --ignored in safeguard analysis
  ```

- **Removed units and vectors** — This comment tells the programmer that a digital unit or vector was commented out of the test, and why. Here is an example:

  ```
  ! IPG: removed unit
  ! "E2 Data tied to Qbar"
  ```

Test Developer Generated Comments:

You can add warnings to the device tests to highlight conditions or instructions. When IPG encounters a warning message, it prints the message on the screen, along with the line number of the digital test where it appears. Here is an example:

```
154: Adjust vector cycle time to
     expected pulse duration.
```

IPG also logs the message in the `ipg/summary` report and the `ipg/details` report.

Analog Functional Tests

IPG does not modify analog functional tests as it does digital tests. If IPG finds a problem with an analog functional tests, such as an inaccessible node, it adds the keyword `nulltest` to the `testorder` listing for that test. This means that resources are not assigned for that test. You can correct the library test and re-run IPG. You can use the `ipg on` statement to run IPG on a single device.

Mixed Tests

IPG treats the analog portion of mixed tests just as analog functional tests; it does not modify them for topology. IPG treats the digital portion of mixed tests just as digital tests. If IPG modifies the digital portion of the mixed test, IPG also flags the test as `comment` in the `testorder` file.

Testplan Generator: Reference

The Testplan Generator (TPG), generates the `testplan` file. The testplan is comprised of a `testmain` that controls the testing of the board, and subroutines that contain the statements to execute the device tests listed in **Table 4-19**.

Table 4-19 Testplan subroutines

Subroutines	Description
Pre_Shorts	executes pre-shorts tests
Shorts	executes shorts and opens testing
Analog_Tests	executes analog in-circuit tests
Check_Power_Sub	Tests that are labeled as `POWER` in the `testorder` file are executed from this subroutine. The `Check_Power_Sub` subroutine is called from the `Setup_Power_Supplies` subroutine.
Digital_Tests	executes digital in-circuit tests
Functional_Tests	executes digital functional tests
Analog_Functional_Tests	executes analog functional tests
Analog_Functional_Tests	executes mixed tests

Testplan Generator Inputs

TPG uses the `testorder` file from IPG, a `testmain` file, and information from the board description file, `board.o`, to generate the testplan.

Running the Testplan Generator

TPG is run by the Develop Board Test function of IPG Test Consultant. You can run this program manually with the `testplan generation` (BT-BASIC) statement.

TPG looks for a `testmain` under the local `ipg` directory, if it does not find one, it uses the standard `testmain`, `$AGILENT3070_ROOT/standard/testmain` (or `abhtestmain` if you specified a cassette fixture in Board Consultant). The available testmains are listed in **Testmain**. If you want to use a different `testmain`, copy it to `ipg/testmain` before running TPG.

When run in the incremental mode, TPG deletes the existing testplan and generates a new testplan. Add all modifications to the new testplan. Instruct TPG to leave the existing testplan and not generate a new testplan by adding the `testplan generation off` statement at the first line of the `testorder` file.

Testplan Generator Outputs

TPG generates the `testplan` file. In the incremental mode, TPG also generates a file called `testplan.diff` that shows the changes made to the testplan.

Testmain

TPG places `testmain` at the beginning of the testplan. `Testmain` controls the testing of your board.

There are four `testmains` provided with the 3070 Series II. They are described in **Table 4-20** on page 4-51.

`Testmain` includes optional features that are easily turned on and off. All features are controlled by usage flags in the `sub Set_Custom_Options`.

Table 4-20 Four `testmains` provided with the 3070 series II

Name	Description
`testmain`	The standard `testmain` under `$AGILENT3070_ROOT/standard/testmain`.
`testmain_panel`	The standard `testmain` for testing panelized boards with PanelTest. It is under `$AGILENT3070_ROOT/standard/testmain_panel`.
`abhtestmain`	This is the `testmain` to use with the Express Fixturing System. You can find this under `$AGILENT3070_ROOT/standard/abhtestmain`.
`abhtestmain_panel`	This is the `testmain` to use with the Express Fixturing System when testing panelized boards with PanelTest. You can find this under `$AGILENT3070_ROOT/standard/abhtestmain_panel`.

Creating Custom Executable Tests: Reference

You can provide custom executable tests for devices that do not have standard or custom library tests. This includes digital in-circuit, digital functional, analog functional, and mixed device tests.

You are responsible for writing custom executable tests; IPG does not help you as it does with library tests.

You can write complete tests or setup-only tests. The setup-only tests include just enough information for the Fixture Generation Software to assign testhead resources. The setup-only tests can be completed later. To save time during the test development process, you can complete the setup-only tests while waiting for your fixture to be built, or even after the test and fixture have been released to production.

To write a custom test we recommend that you copy a library test from the standard 3070 Series II and 3 libraries to the appropriate directory in your local board directory. Sometimes you can find a test for a device that is similar to the device you want to test and then modify a copy of it to meet your needs. This simplifies the process and helps you maintain the structure of the library tests. In addition to the standard libraries, there are analog functional test templates under `$AGILENT3070_ROOT/library/templates`. You can copy these templates to your test directory and edit them to test your particular devices.

Any executable tests (setup-only or complete) must be written before generating the test requirements (first compile).

Executable tests, including setup-only tests, are compiled to generate requirements (.r) files in Task 4; they are compiled again, in Task 7, to generate object (.o) files.

Devices for which you write executable tests should be entered in Board Consultant with the Pin Library Entry form or the Node Library Entry form, and must be specified as `not tested`. You can optionally specify a part number. You must also make a manual entry in the `testorder` file for these devices to label them as `PERMANENT`.

Note that custom executable tests do not use Agilent SAFEGUARD.

Custom Digital In-Circuit Executable Tests

To create custom digital in-circuit executable tests:

a Create the `digital` test directory for digital tests if it does not exist. For example:

```
create dir "digital"
```

b Write the custom executable tests or copy and modify standard library tests.

Custom Digital Functional Executable Tests

This description outlines the steps necessary to create custom digital functional executable tests. See **Figure 4-2** on page 4-54 for the file structure of a digital functional test.

The main difference between digital in-circuit and digital functional testing is that digital functional testing allows backtracing of internal nodes, while digital in-circuit testing does not.

Note that device clusters must be entered in Board Consultant. You should enter internal devices for your functional tests to ensure that correct backtrace files are generated.

To create custom digital functional executable tests:

a Create the functional test directory for digital functional tests if it does not exist.

```
create dir “functional”
```

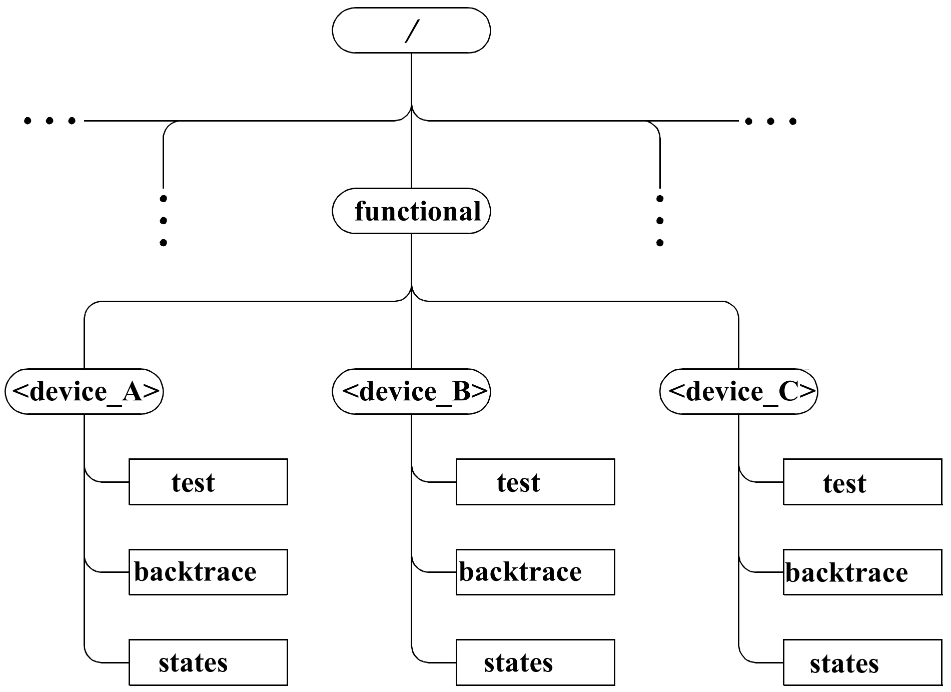
b Create the individual directory under the functional directory for each digital functional test.

```
create dir "functional/ram"
```

c Write the custom executable test and backtrace files, or copy and modify standard library test.

- `functional/ram/test`
- `functional/ram/backtrace`

Figure 4-2 File structure of a digital functional test



Custom Analog Functional Executable Tests

Use analog functional tests to test analog devices with power applied to the board under test. Analog functional devices are devices such as operational amplifiers (you can also test these with a digital test), voltage regulators, and oscillators.

To create custom analog functional executable tests:

a Create the `analog` test directory, if it does not exist.

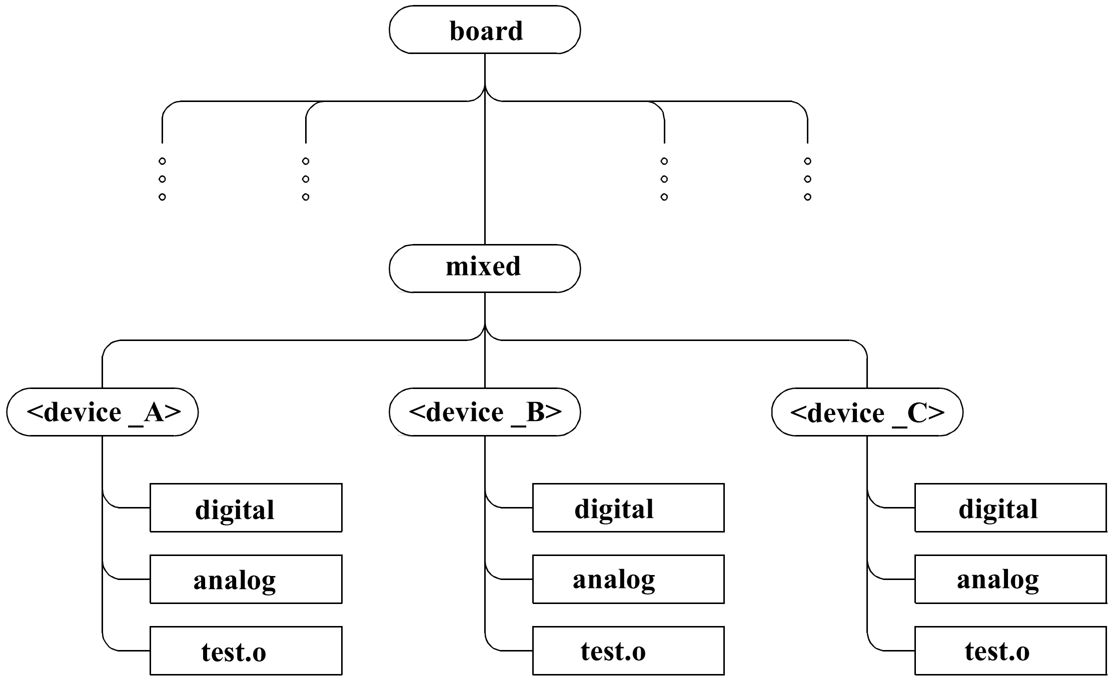
b Write the custom executable tests, or copy and modify standard library tests.

Custom Mixed Executable Tests

Use mixed tests for devices that are part digital and part analog, such as digital to analog converters.

Figure 4-3 shows the file structure of mixed tests. Each mixed test is a directory that includes an analog test source file (`analog`) and a digital test source file (`digital`).

Figure 4-3 File structure of mixed tests



To create custom mixed executable tests:

a Create the mixed directory, if it does not exist.

```
create dir "mixed"
```

b Create the individual mixed test directory.

```
create dir "mixed/d_to_a"
```

c Copy and edit, or create the analog and digital portions of the mixed test.

```
"mixed/d_to_a/analog"
"mixed/d_to_a/digital"
```

NOTE

Note that mixed executable tests cannot use SAFEGUARD. Mixed tests pass control between the digital and analog subsystems which makes them time-indeterministic. You have to manually disable all upstream devices to prevent damage.

Edit the Testorder File

Because executable tests do not use IPG, you need to manually add your custom executable tests to the `testorder` file. To add your custom executable tests to the list in the `testorder` file, follow these three steps:

a Load the testorder file.

```
load "testorder"
```

b Add each executable test with the permanent keyword.

```
test analog powered "U1"; permanent
test mixed "d_to_a"; permanent
```

c Re-store the testorder file.

```
re-store "testorder"
```

The `permanent` keyword ensures that the line in the `testorder` file, and the associated test, does not get changed by IPG. Now you have a `testorder` file that declares your executable tests.

NOTE

The `testorder` file and its syntax are discussed in Chapter 3, **The Testorder File** in *Data Formats*.

Fixture Generation Software: Reference

The software runs in one of two modes: **new** for generating a new fixture and **incremental** for making changes before or after your fixture has been built. If you make changes, re-run Develop Board Test to update all necessary files.

The fixture generation process, as shown in **Figure 4-4**, includes four sections described in **Table 4-21** on page 4-60.

Figure 4-5 on page 4-62 shows the Fixture Generation Software structure. The `board_xy.o` and `board.o` files are inputs to the software. These files are the compiled object versions of the `board_xy` and `board` source files. You can create the source files by running the CAMCAD, or by editing your `board` and `board_xy` files using Board Consultant or BT-BASIC.

The Summary and Details Reports are stored in the `fixture/summary` and `fixture/details` files. Do not confuse these files with the `ipg/summary` and `ipg/details` files. The Summary and Details Reports discussed in this chapter refer to fixturing, not IPG.

Note that the `fixture/summary` and `fixture/details` files are actually created by Board Placement. They are indicated under Fixture Tooling because that is where they are completed. They are started by Board Placement, then Probe Select, MPA, and Fixture Tooling add information to them (these lines are not shown).

Figure 4-4 The four sections of the fixture generation process

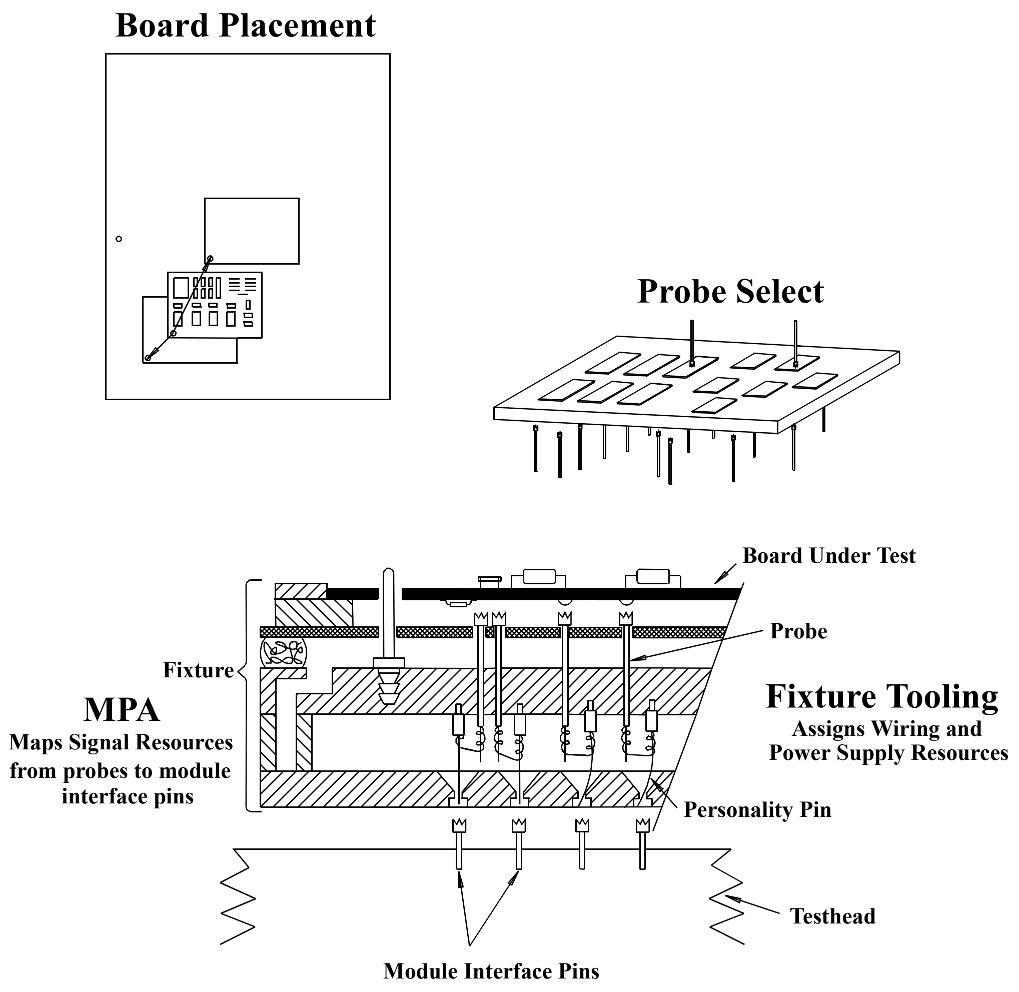


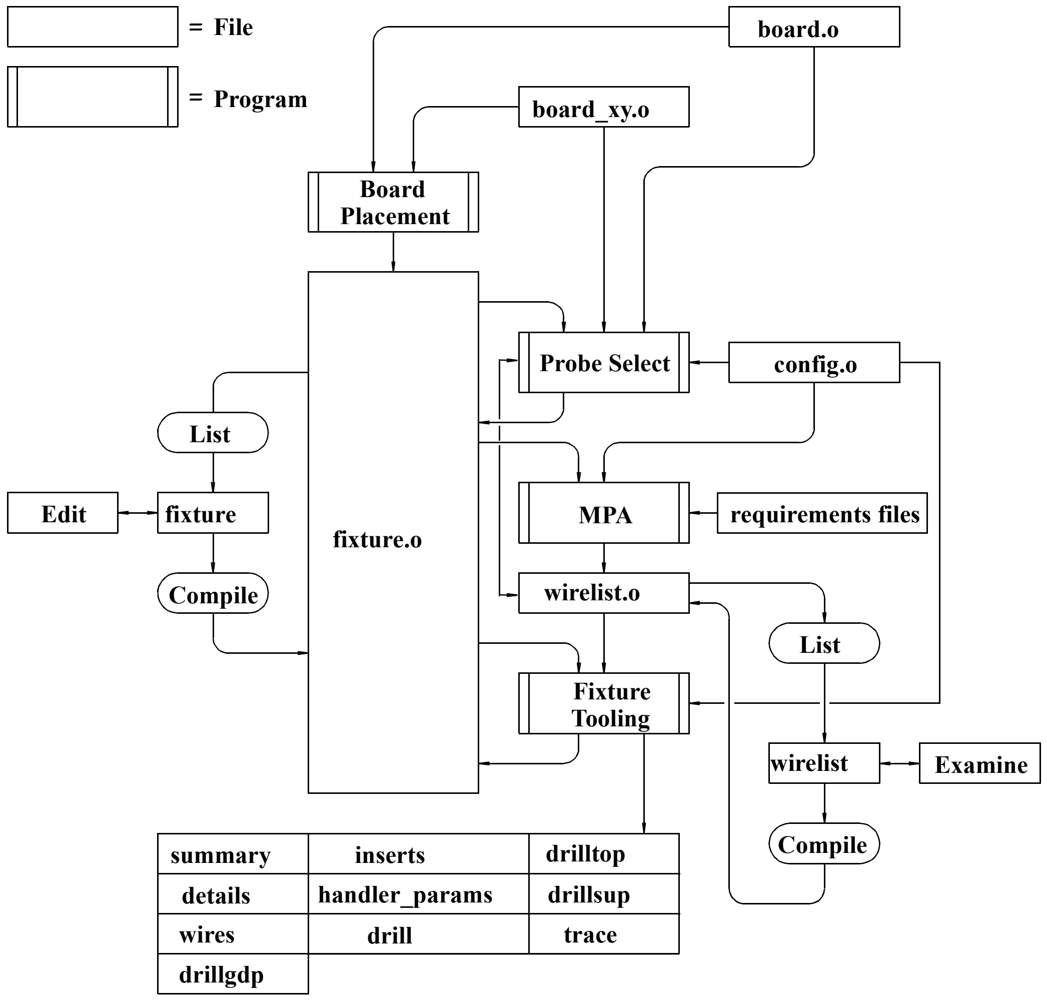
Table 4-21 Fixture generation process

Section	Description
Board Placement	Creates the fixture file system (except `wirelist.o`) if it does not already exist. If you manually specify a placement, Board Placement uses your specification. Otherwise, it tries to place the board in the lower left corner of the fixture so the placement is compatible with the Express Fixturing System. You can also instruct Board Placement to position the board over the center of the testhead configuration; this is not necessarily on the center of the fixture.
Probe Select	Chooses the best type of probe and the best location, physically and electrically, to contact each node on the board under test.

Table 4-21 Fixture generation process (continued)

Section	Description
Module Pin Assignment (MPA)	maps and assigns signal resources (through module interface pins and test probes) to nodes. It tries to assign resources close to each node to minimize wire lengths, especially on critical nodes. MPA creates the `wirelist.o` file.
Fixture Tooling	assigns power and ground resources (through module interface pins and test probes) to nodes. Organizes the wiring connections from all resources to the fixture probes as listed in the `wirelist.o` file and produces files and reports that instruct you how to build the fixture. The types of reports and files produced depends on the type of fixture you are using. These include: • Wires Report • Inserts Report • Trace Report • Summary Report • Details Report • Difference Report • `drill` file • `drillsup` file • `drilltop` file • `drillgdp` file • `handler_params` file

Figure 4-5 Fixture generation software flow diagram (for **new** mode)



Board Placement

Board Placement creates the fixture file structure (except the `wirelist.o` file), if it does not already exist. Board Placement determines the fixture part number and records it in the `fixture.o` file.

Board Placement automatically determines the board's locality on the test fixture and specifies the location in the `fixture.o` file. The placement is specified as the X and the Y direction offsets of the board origin from the bank 2 fixture origin (the bank 1 drill reference origin is 142500 tenth-mils in the X direction from the bank 2 fixture origin). The placement also includes a rotation specification that causes the board to be rotated the number of degrees specified. A positive rotation causes the board to be rotated counter-clockwise; a negative rotation causes the board to be rotated clockwise. The rotation is applied to the board first, then the X and Y offsets are applied. For example, the placement specification (in the `fixture.o` file) for the board in **Figure 4-6** on page 4-64 could be:

```
PLACEMENT 30362, -59276 0.0
```

This would position the board's origin 3.0362 inches to the right and 5.9276 inches down (toward the operator) from the fixture origin. There would be no rotation.

Unlike all other coordinates in the `fixture.o` file, placement coordinates are not offset to positive values when added to the `fixture.o` file.

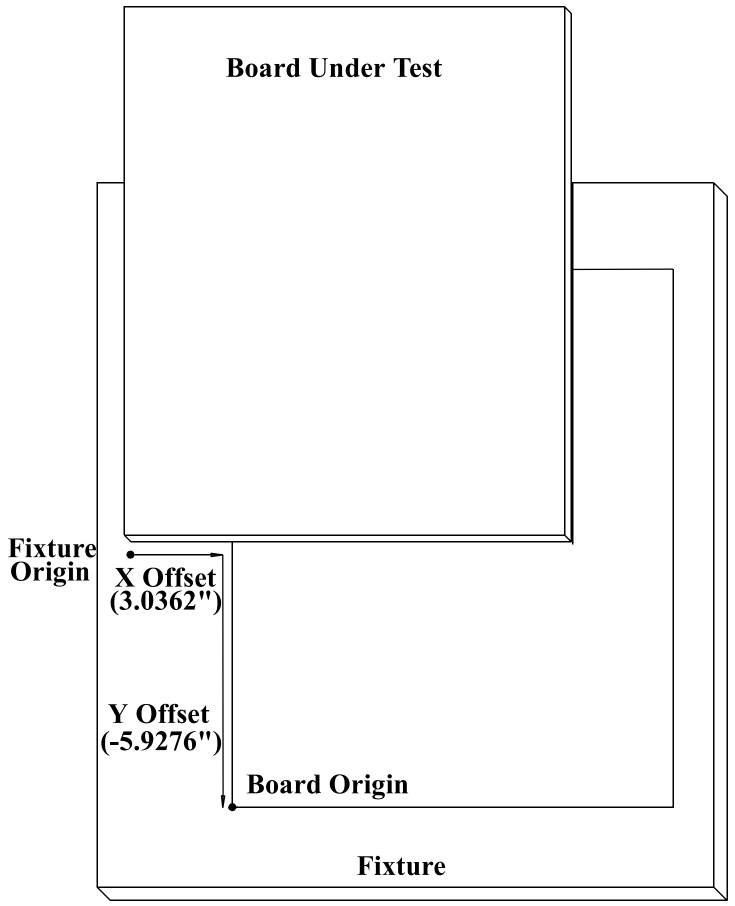
If the board is physically too large to fit on the fixture, Board Placement generates the error message:

```
A board of this size will not fit on the
fixture.
```

You can manually instruct Board Placement where to place the board on the test fixture as described earlier in this chapter. If you do not specify a location, Board Placement places the board in one of three positions on the fixture:

- Near the lower left corner of the fixture, with the longest edge down (toward the operator). This placement is required for the cassette fixtures and facilitates the possibility of converting an express fixture to a cassette.
- Centered over the testhead pin card configuration (not necessarily centered on the fixture).
- Within the outline of the Agilent No-Wire Technology fixture Printed Circuit Board.

Figure 4-6 Board placement



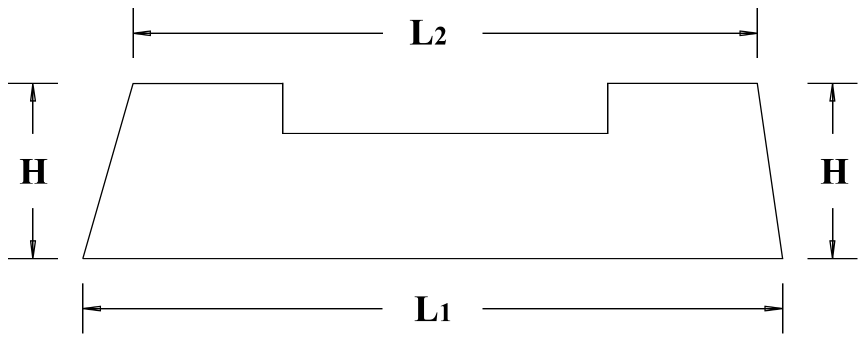
Board Placement for Cassette Fixtures

For express and cassette fixtures, Board Placement places the board in the lower left corner with the longest side down (toward the operator).

After determining the board's position on the fixture, Board Placement checks the shape of the board for compatibility with the board handler and cassette fixture. To be compatible, the board's shape must fit two criteria (refer to **Figure 4-7**):

- The top-to-side aspect ratio (L_2 / H) cannot be less than 0.75.
- The bottom-to-side aspect ratio (L_1 / H) cannot be less than 0.75.

Figure 4-7 Board shape check



If the board meets these criteria, Board Placement positions the board such that the longest edge is against the board handler's drive rail.

If the board does not meet these criteria, Board Placement rotates the board and checks the shape again.

NOTE

If the board does not meet the shape criteria with any rotation, Board Placement issues this warning message if an Express fixture is specified: “This board cannot be placed for automatic board handler compatibility.”

This placement is not compatible with the automatic board handler.

For the SimPlate fixture, Board Placement does not require board outline coordinates, board placement specifications, or tooling pin hole locations. If this information is specified in the `board_xy.o` file, Board Placement copies it to the `fixture.o` file, but it is not used.

Board Placement for Centering the Board

If Board Placement is run with the `center` option, it positions the board over the testhead pin card configuration. Note that this is not necessarily centered on the test fixture.

You can specify the `center` option in one of two ways:

- Run Board Placement with the `board placement` (BT-BASIC) statement and include the `center` option.
- Run Board Placement with Develop Board Test and include the `center` option in the `.hp3070` file.

Board Placement for Agilent No-Wire Technology Fixtures

If the **Fixture Type** in IPG Global Options is set to **No-Wire**, Board Placement will place the board within the outline of the No-Wire fixture PCB. The Device Under Test must be completely above the fixture PCB to allow the DUT receptacles to contact the PCB.

How to Run Board Placement

You can run Board Placement with the Develop Board Test function of IPG Test Consultant or with the BT-BASIC `board placement` and `board placement on` statements; this is explained earlier in this chapter.

Board Placement runs in one of two modes, **new** or **incremental**. In **new** mode, Board Placement creates a new fixture. In **incremental** mode, it makes changes to the fixture building reports after the Fixture Generation Software has been run, whether or not the fixture has been built.

Board Placement in New Mode

If the `fixture.o` file does not exist, Board Placement runs in new mode. In the new mode, Board Placement:

- creates the fixture file system
- looks for a board placement specification in the `board_xy.o` file, if there is no placement specification, Board Placement determines the board's location on the test fixture
- looks for board keepout areas specified in the `board_xy.o` file
- records the placement and board keepout areas in the `fixture.o` file

Board Placement in Incremental Mode

If the `fixture.o` file does exist, Board Placement runs in incremental mode and checks the board fit: it checks to make sure that the board outline coordinates, tooling pin hole locations, or keepout areas have not changed.

In incremental mode, Board Placement appends information to existing files; it does not create new files. If the board outline or tooling holes have changed, Board Placement warns you that the changes have been recognized, but ignored. If you change your board outline or tooling holes and you have not built your fixture, you should unlink the `fixture.o` and `wirelist.o` files to run the Fixture Generation Software in new mode.

Board Placement Inputs

Board Placement uses the information described from the files shown in **Table 4-22**.

Board Placement Outputs

Board Placement adds information to the `fixture.o`, `fixture/summary`, and `fixture/details` files. This information is described in **Table 4-23**.

Table 4-22 Board placement inputs

File	Description
`board_xy.o`	Provides the board outline coordinates. Board Placement also uses placement specifications and keepout areas if there are any. If a cassette fixture was specified, Board Placement looks for the existence of LONG nodes to determine which type of Express Cassette fixture to use: floating or fixed top plate.
`board.o`	Provides the size and type of fixture. If a cassette fixture was specified, Board Placement also looks for the Top Probes Allowed flag to determine which type of Express Cassette fixture to use: fixed or floating top plate.
`config.o`	Provides the location and type of the cards in the testhead.

Table 4-23 Board placement outputs

File	Description
`fixture.o`	The placement is specified in the `fixture.o` file, and includes the board outline coordinates, tooling pin hole and locations, board placement specifications, fixture part number, and fixture options. Board Placement also adds keepout areas to the `fixture.o` file if any are specified in the `board_xy.o` file.
`fixture/summary`	Error and warning messages about compatibility with the Express Cassette fixture. Board Placement also confirms if the board fits on the fixture.
`fixture/details`	The same information as in the `fixture/summary` file, but with much more detail. This file can also contain a Difference Report when fixture generation software is run in incremental mode.

Board Placement and Multiple-Board Fixtures

For multiple-board fixtures, Board Placement treats the addition of subsequent boards as changes to the `fixture.o` file, but processes each additional board in new mode. Board Placement tries to place up to two boards on a standard (single-bank) fixture and up to four boards on a large (two-bank) fixture. It tells you if it fails to place any board.

Probe Select

The Probe Select process illustrated in **Figure 4-8** determines the best type of probe for each node on the board and finds the best location, both physically and electrically, to contact each node.

Probe Select uses information from the `board_xy.o`, `board.o`, and `config.o` files. It provides output information to the `fixture.o`, `fixture/summary`, and `fixture/details` files.

How to Run Probe Select

You can run Probe Select with the Develop Board Test function of IPG Test Consultant or with the BT-BASIC `probe selection` and `probe selection on` statements; this is explained earlier in this chapter.

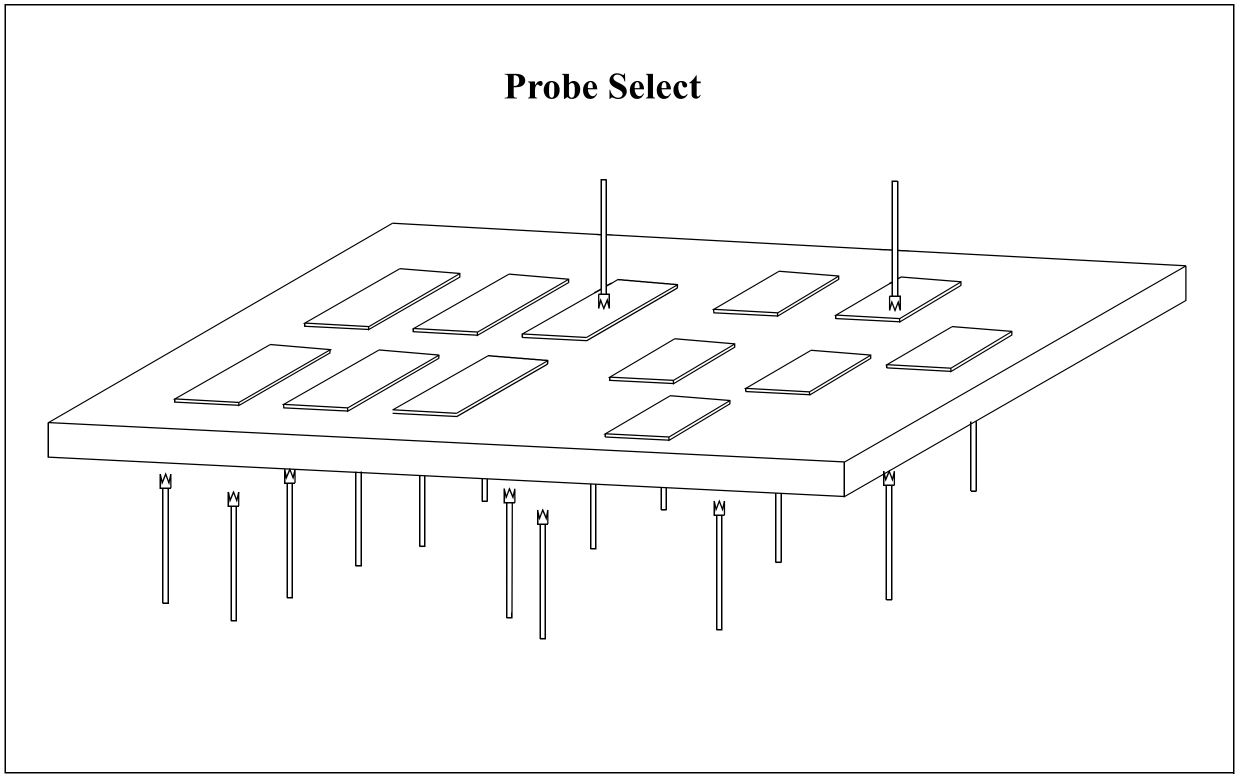
Probe Select runs in one of two modes, **new** or **incremental**. If the `fixture.o` file has no probes listed, Probe Select assumes the **new** mode. If the `fixture.o` file has one or more probes, Probe Select assumes the **incremental** mode. Probe Select runs once for each fixture, whether it is a single-board or multiple-board fixture.

Probe Select in New Mode

In **new** mode, Probe Select:

- Assigns probes to every node possible.
- Tries to use 100-mil heavy spring force probes, especially on critical nodes, rather than 100-mil light spring force, 75-mil, or 50-mil probes. (Does not apply to SimPlate fixture.)
- Tries to use bottom probes rather than top probes on cassette fixtures. (Does not apply to SimPlate fixture.)
- Distributes probes across the board to avoid dense areas of probes. (Does not apply to SimPlate fixture.)
- Avoids placing probes near the board edge or tooling pin holes. (Does not apply to SimPlate fixture.)
- Places probes as close as possible to critical device.pins.
- Determines the best location to probe each node based on electrical considerations and places the probe as close to that location as possible.
- Assigns a partially drilled hole for all unconnected device.pins and scarce resources for future use.
- Avoids blocking GP Relay, Control Card, and AccessPlus Card resources. (Does not apply to SimPlate fixture.)
- Tries to select all locations that are labeled as `MANDATORY`. (Does not apply to SimPlate fixture.)
- Tries to select all locations of a `MANDATORY GROUP` as probe locations. It also identifies these probes as `GROUPS` in the `fixture.o` file. If Probe Select cannot probe one of the paired locations of a GROUP, it issues a warning. (Does not apply to SimPlate fixture.)
- Avoids placing personality pins, probes, and transfer pins in a keepout area. If the only location for probing a node is within the keepout area, Probe Select declares the node inaccessible and issues a warning. (Does not apply to SimPlate fixture.)

Figure 4-8 Probe select



The SimPlate fixture does not use location information (the exception to this rule is edge-connector-only testing).

Probe Select calculates the number of ground probes needed depending on the type of fixture used. The number of ground probes needed is shown in **Table 4-24**.

Table 4-24 Required ground probes

Fixture	Probes Required
XG-50 fixtures	One ground probe for every five digital probes.
Express fixtures	One ground probe for every 32 probes.
SimPlate fixtures	One ground probe for every 32 probes plus one ground probe for every three digital probes.

Probe Select in Incremental Mode

In **incremental** mode, Probe Select:

- compares the `fixture.o` and `board.o` files to detect topological changes
- compares the `fixture.o` and `wirelist.o` files for node-to-brc changes

To minimize fixture rework, Probe Select manages probing changes by choosing probing locations in this order:

- Looks for an existing probe that contacts the node.
- Looks for a partially drilled hole (alternate) in which to add a probe to contact the node.
- Chooses a new probe location that is not blocked.

If you do not want to drill new holes in the fixture probe plate, you can declare the nodes as NO_PROBE and run the Develop Board Test function of IPG Test Consultant. This regenerates the fixture files and device tests, considering the NO_PROBE nodes inaccessible.

If no existing locations are available (probes inserted or partially drilled) and no new locations can be used, Probe Select declares the new or changed node as inaccessible.

Upon completion of a run in incremental mode, Probe Select:

- lists the number of new probes needed in the `fixture/summary` file.
- specifies the location of new probes, or of probes that need to be inserted into existing alternate holes, in the `fixture/details` file.
- adds location information about the new probes to the `fixture.o` file.

Probe Select Inputs

Probe Select processes information from the files described in **Table 4-25**.

Table 4-25 Probe select inputs

File	Description
`board.o file`	This file contains global options information, such as fixture type and size, test strategy, node connections, power supply connections, and critical pins.
`config.o file`	This file is the board configuration file found in your local board directory. It describes the hardware available in the testhead, such as: • modules and cards installed, • DC DUT power supplies installed. Do not confuse the board configuration file, `config`, with the system configuration, also named `config`.
`board_xy.o file`	X-Y locations of the device.pins, tooling holes, and board outline. This file is optional for the SimPlate fixture (so that NO-PROBE nodes can be declared), unless the test strategy specified is `edge connector only`.
`fixture.o file`	The `fixture.o` file provides Probe Select with the board placement.
`wirelist.o file`	For incremental mode.

How Test Strategy Affects Probe Select

The test strategy specified in the `board.o` file determines some of the actions performed by Probe Select. You can specify the test strategy, `combinational` or `edge connector only`, in Board Consultant. Table 4-26 explains the differences in Probe Select for each test strategy.

You should also note that for this strategy, a `board_xy.o` file is necessary for all fixtures, including the SimPlate fixture. Because actual X-Y locations are irrelevant to the SimPlate fixture, the `board_xy.o` syntax allows you to use asterisks (**) for X-Y coordinates.

Probe Select considers all nodes to be inaccessible, except those that have locations in the `board_xy.o` file; this is true for all fixture types.

Table 4-26 How test strategy affects probe select

Strategy	Description
combinational	For this test strategy, Probe Select: ■ issues warnings for nodes that have no locations in the `board_xy.o` file (except for the SimPlate fixture) ■ lists all NO_PROBE nodes in the `fixture/details` file ■ issues inaccessible node warnings for nodes that: • have only EXTRA locations • have only top-side locations when top-side probing has been turned off • or have all locations labeled NO_PROBE A `board_xy.o` file is optional for the SimPlate fixture. You can use a `board_xy.o` file to specify NO_PROBE nodes. Probe Select considers all nodes to be accessible on an SimPlate fixture except those labeled NO_PROBE in the `board_xy.o` file.
edge connector only	Contrary to the combinational test strategy, for edge connector only testing, Probe Select does not issue the warnings or list NO_PROBE nodes at all.

Module Pin Assignment

MPA maps and assigns test resources to nodes. It tries to assign resources close to each node to minimize wire lengths, especially on critical nodes. MPA cannot minimize wire lengths on the SimPlate fixture because location information is not available.

MPA creates the `wirelist.o` file, and adds information to the `fixture/summary` and `fixture/details` files.

MPA must occasionally assign more than one wire to a node to avoid multiplexing conflicts. No conflicts should occur with critical pins and nodes because they are assigned first. Exactly how MPA multiplexes resources primarily depends on the types of cards used.

Figure 4-9 demonstrates Module Pin Assignment.

MPA runs once, on all boards, for a multiple-board fixture.

How to Run MPA

You can run MPA with IPG Test Consultant or with the BT-BASIC `module pin assignment` statement.

MPA in New Mode

In new mode, MPA assigns test resources to nodes and lists this information in the `wirelist.o` file.

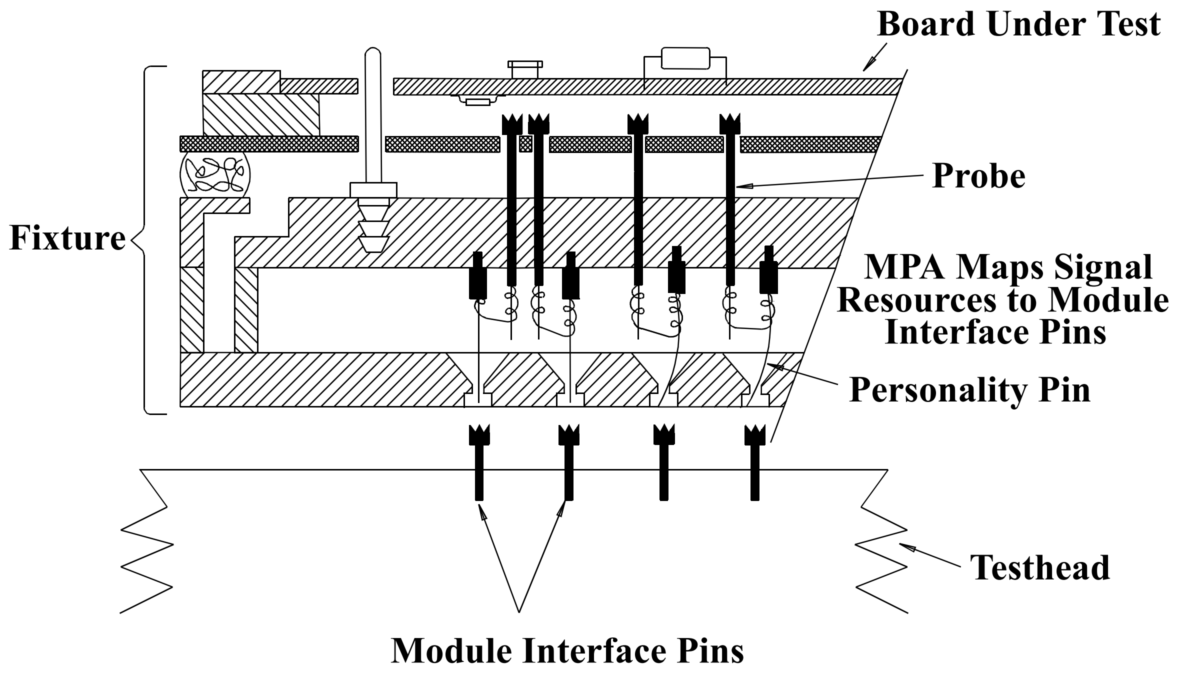
The nodes to which MPA first assigns resources have the best possibility of acquiring short fixture wires because most resources are available. As more resources are assigned to nodes, fewer resources are available, so fixture wires tend to become longer. To promote the assignment of short fixture wires on the nodes that need short wires, MPA assigns resources to nodes in the following order.

1. clock nodes
2. precision analog nodes (nodes used with external instruments, the Time Interval Counter detector, or access instruments marked `direct`)
3. digital functional nodes
4. digital nodes on large devices
5. digital nodes on small devices
6. hybrid nodes
7. analog nodes
8. shorts nodes
9. GP relay nodes

NOTE

You can command MPA to assign particular nodes first to promote short fixture wires for them by specifying them as CRITICAL as explained in Chapter 3, **Creating Board Information.**

Figure 4-9 Module pin assignment



You should override MPA's prioritized assignment process if one of the following circumstances is true.

- You have a node used for a high-speed clock in an analog functional test. Unless you label this node as a clock, MPA assigns resources to it after all hybrid nodes. You can specify the node as CRITICAL and MPA assigns resources to it before any other node.
- You have a test device that meets the following criteria:
  - The device has edge-sensitive inputs.
  - The edge-sensitive inputs are not overdriven.

- The device upstream from the edge-sensitive inputs is in a high-impedance state.

NOTE

Specifying CRITICAL also affects Fixture Tooling, which assigns ground wires to corresponding pin cards to encourage the shortest overall path possible from the device to the pin cards. Refer to the section on **Fixture Tooling** on page 4-80 for details.

MPA in Incremental Mode

In incremental mode, MPA reads the `wirelist.o` file to preserve the existing fixture wiring. MPA also updates the `wirelist` source file, if one exists in the board directory. In the case of a fixture overflow, MPA updates only the `wirelist` source and not the `wirelist.o` object file. In incremental mode, MPA gets a list of drilled, extra resources from the `fixture.o` file.

In incremental mode, MPA preserves as much of the existing wiring as possible. New or changed tests that have a requirements (.r) file are evaluated; all other tests are preserved. MPA assigns what resources it can and adds this information to the Summary and Details Reports. If MPA is run with the `drilled` option, it is restricted to using the EXTRA resources listed by Fixture Tooling. These extra resources are either drilled and have receptacles inserted or they are just partially drilled. If MPA is run without the `drilled` option, it is not restricted.

MPA Inputs

The inputs to MPA are shown in **Table 4-27**.

Table 4-27 Module pin assignment inputs

File	Description
requirements (`.r`) files	Provide a list of the required resources for each device test.
`board.o` file	Provides a list of GP Relay connections.
`testorder` file	Provides a list of the requirements files.
`config.o` file	Identifies the hardware available in the testhead. This information includes: • modules and cards installed • DUT power supplies installed • functional ports installed • connections to functional ports • access ports installed • connections to access ports
`fixture.o` file	Provides a list of critical nodes, blocked pins, and probe locations.

MPA Outputs

The MPA program creates the `wirelist.o` file and adds information to the `fixture/summary` and `fixture/details` files. The MPA Outputs are described in **Table 4-28**.

NOTE

You can find syntax descriptions of the statements in the `wirelist.o` file in Chapter 4, **The Wirelist File** in *Data Formats*.

Table 4-28 Module pin assignment outputs

File	Description
`wirelist.o` file	MPA lists the wiring from node to brc for each test. It also specifies the type of resource for each wire. Note that only the wirelist object file, `wirelist.o`, is maintained in the local directory. In the case of a fixture overflow (more nodes than resources), MPA writes only a `wirelist` file, not a `wirelist.o` file.
`fixture/summary` file	MPA notes the experience level at which it was run (standard, advanced, or expert). MPA lists the average wire length for precision nodes. Precision nodes are all nodes except most analog and GP relay wires. The only analog nodes included are those used for external instruments and the time interval counter (TIC). This file also shows the number of brc's used and calculates the efficiency as nodes/brc's. Note that there is no wire length information for the SimPlate fixture.
`fixture/details` file	This file contains all the information of the `fixture/summary` file plus a list of all precision node wires that are longer than six inches. This list includes the brc, node name, and the type of resource.

Resource Restrictions

There are restrictions on which resources can be used for particular tests and on specific nodes. Altering files that control these restrictions could cause multiplexing conflicts, and the tests might not work.

The resource **general** can be used in any test and on any node, but it is not multiplexed.

The restrictions for all other resources are described in **Table 4-29**. In the descriptions of the resource restrictions, the definitions shown in **Table 4-30** on page 4-80 are used.

Table 4-29 Resource restrictions

Test	Resources Allowed in One Test	Resources Allowed on One Node
Analog In-circuit Tests	s & i & b & (g)* & (l|gl) & (a | aux)	(s & a) | (i & b) | (g & l) | gl | aux
Analog Functional Tests	detector low & detector high & source & ext1-8 & aux & (clock receive frequency | receive frequency| receive interval) & g	Only one of any given resource
Mixed Tests	detector low & detector high & source & ext1-8 & aux & (drive)* & trigger & trigger & trigger & (receive)* & clock receive & (clock | fast clock) & (clock receive frequency | receive frequency | (receive interval & receive interval)) & g	aux | detector high | detector low | source | ext1 | ext2 | ext3 | ext4 | ext5 | ext6 | ext7 | ext8 | (drive | clock | fast clock) & (clock receive | (receive & clock receive frequency) | receive frequency | receive interval) | g
Digital In-circuit Tests	(drive)* & trigger & trigger & trigger & (receive)* & clock receive & (clock | fast clock)	(drive | clock | fast clock) & (clock receive | (trigger & receive))
Digital Functional Tests	(drive)* & trigger & trigger & trigger & (receive)* & clock receive & (clock | fast clock) & receive fixed) | receive internal	(drive | clock | fast clock) & (trigger & receive) | (clock receive | trigger) | (trigger & receive fixed)) | receive internal
Shorts & Opens Testing	(g)*	g

Table 4-30 Resource restriction definitions

Term	Definition
A	specifies `A` zero or one times
(A)*	specifies `A` any number of times
A & B	specifies `A` and `B` can be present together
A | B	specifies `A` or `B` can be present, but not both

Fixture Tooling

Fixture Tooling assigns power and ground resources (through module interface pins and test probes) to nodes. It also assigns and records the wiring connections from module interface pins to nodes as mapped by MPA and listed in the `wirelist.o` file. The wiring includes pin-to-probe and pin-to-pin connections. Fixture Tooling produces files and reports that instruct you how to build the fixture. The types of reports and files produced depend on the type of fixture you are using.

Fixture tooling does not create drill files for the SimPlate fixture and the other reports specify device pins instead of X-Y locations. Fixture Tooling runs once on all boards of a multiple-board fixture.

Figure 4-10 presents a diagram of Fixture Tooling.

How to Run Fixture Tooling

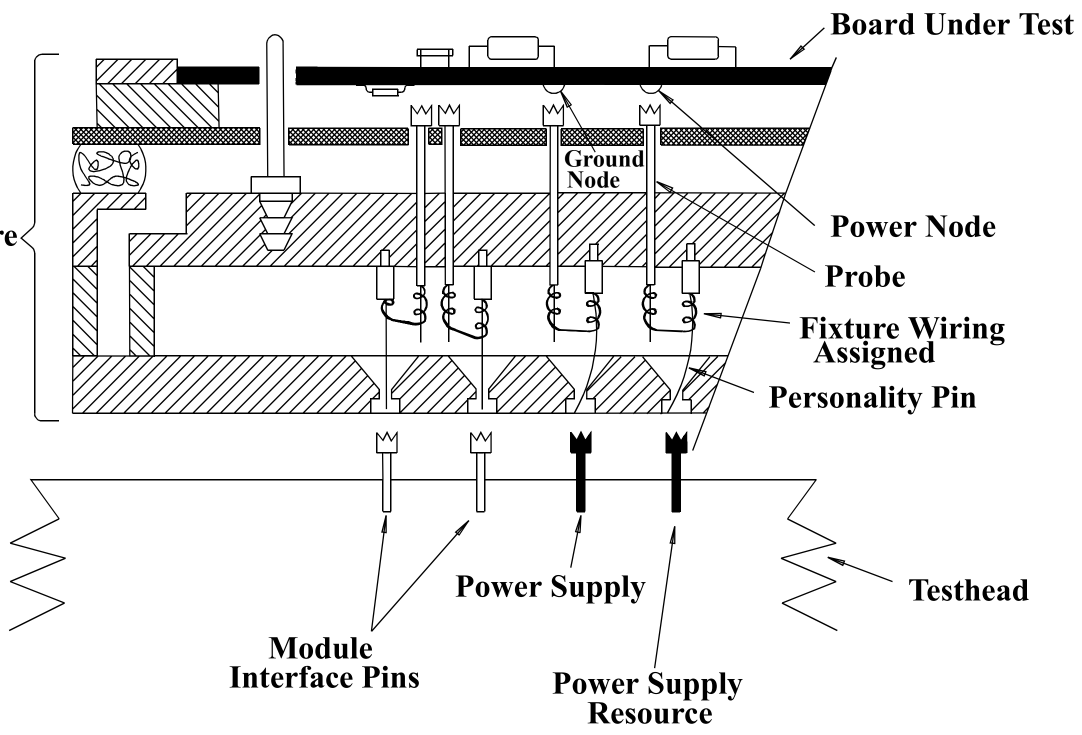
You can run Fixture Tooling with the Develop Board Test function of IPG Test Consultant or with the BT-BASIC `fixture tooling` statement.

Fixture Tooling in New Mode

In new mode, Fixture Tooling writes the fixture wiring and building instructions, and assigns power and ground connections to the board under test. The main tasks executed by Fixture Tooling include:

- Assign the wiring connections to the fixture probes from all resources as listed in the `wirelist.o` file. This includes pin-to-probe and pin-to-pin wiring. Fixture Tooling also produces the files and reports that instruct you how to build the fixture.

Figure 4-10 Fixture tooling



100-mil and 75-mil receptacles hold three wirewraps; 50-mil receptacles hold one wire. If more wires need to be attached to the receptacle than can fit, Fixture Tooling calls for pin-to-pin wiring. That is, instead of wiring every personality pin to the probe receptacle, some of the personality pins are wired to each other (daisy-chained), then to the probe receptacle.

For the AccessPlus Card, Fixture Tooling avoids pin-to-pin wiring by assigning all AccessPlus Card connections prior to assigning other pin card connections. Fixture Tooling assigns the same length, color, and gauge wires to both nodes specified in a group. Fixture Tooling lists the wires that are assigned to groups in the Summary and Wires Reports.

For the SimPlate fixture, Fixture Tooling allows only one wire (two wires for remote sensing) to be wrapped to the probe receptacle. This minimizes the number of wires crossing the fixture hinge.

- Assign power and ground resources (through module interface pins and test probes) to nodes. Assign and record the DUT power supply wires. Assign resources from the ground nodes to Pin Card and AccessPlus Card grounds, and ASRU and Control Card switched grounds and safety disables.

  Fixture Tooling provides at least one power wire for each amp of expected current required by the board under test, up to a maximum of six power wires for the Agilent 6624 power supply, 12 for the Agilent 6621 power supply, plus one power sense wire.

  Because high-speed digital testing might require short ground wires, Fixture Tooling tries to promote short ground wires on the pin cards connected to nodes labeled as CRITICAL. It does this by assigning ground wires in the following order:

  - Power supply grounds
  - Pin card grounds for pin cards connected to CRITICAL nodes
  - All other pin card grounds

  If there is more than one pin card connected to CRITICAL nodes, Fixture Tooling assigns a ground wire to each card in a rotating manner until all unblocked ground pins on each card are connected. This gives each card the best possibility of having one ground wire that is no longer than 2.00 inches (60mm).

  Fixture Tooling adds a table of ground-wire lengths, for each pin card that is connected to a CRITICAL node, to the `details` file. Each ground wire length greater than 2.00 inches (60mm) is marked with a percent sign (%). Fixture Tooling provides this table only in the new mode, not in the incremental mode.

- Assign and record the wires for the fixture autofile pins and fixture enable pins. If you specified an autofile in Board Consultant, Fixture Tooling uses it; otherwise, it assigns the first available autofile starting with 4094 and working toward 11. When choosing an autofile, be sure you do not specify one that is already used.

- Assign and record the wires for connecting the G bus and L bus across modules. These are the only buses that can be connected across modules.

- Determine and record the extra personality pins to be drilled (or partially drilled), and receptacles inserted.
- Produce a series of fixturing reports and files. The reports generated depend on the type of fixture you are using. For more information about the fixture building reports and files, refer to **Reading the Fixture Files and Reports** on page 4-93.

Fixture Tooling in Incremental Mode

In incremental mode, Fixture Tooling re-uses as many existing wires as possible, and does not add extra resources. It tries to minimize the number of changes it makes to the fixture files and reports. Reports and files are completely regenerated, and Fixture Tooling appends a Difference Report to the Details Report that explains the necessary wiring changes to the fixture.

Fixture Tooling Inputs

There are three inputs to the fixture tooling process as shown in **Table 4-31**.

Table 4-31 Fixture Tooling Inputs

File	Description
wirelist.o file	Provides the association of module interface pins, in brc format, to node names.
fixture.o file	Provides: • **options** fixture type, fixture size, fixture part number, inaccessible nodes and probes, and board outline coordinates. • **probes** location of the probe on the node (or device.pin), X-Y location of the probe, alternate probe points available on nodes, and the type of probe used. • **placement** the position of the board on the fixture.
config.o file	Is used to decide: • Autofile wiring • Fixture Enable wiring • Safety disable wiring • DUT power supply wiring • What extra resources to drill • The G and L bus connections across testhead modules

Fixture Tooling Outputs

Fixture Tooling creates or modifies the reports and files shown in **Table 4-32**

Table 4-32 Fixture tooling outputs

File	Description
`fixture.o`	Fixture tooling reads the `fixture.o` file, then adds pin, and wire information to it, and replaces it with a new `fixture.o` file.
`fixture/summary`	Fixture tooling adds fixture component information to the `fixture/summary` file.
`fixture/details`	Fixture tooling adds resource use information and a table of ground-wire lengths for pin cards that are connected to CRITICAL nodes. The `Difference` file is appended to `fixture/details` for modifying an existing fixture.
`Reports` and `Drill` Files	Fixture tooling creates reports and drill files needed to build the fixture. Which reports and files it creates depends upon the type of fixture you are using. The reports and files include: • Wires Report • Inserts Report • Trace Report • Drill Files — `drill`, `drilltop`, `drillsup`, and `drillgdp`. • Difference Report (appended to the Details Report) • `handler_params` file (for the EFS)

Special Considerations for XG-50 Fixtures

This section describes the steps required during the test development process when using an Agilent XG-50 fixture. It also explains the guidelines followed by Fixture Generation Software when an XG-50 fixture is specified. If any of these guidelines cannot be met, the software generates a warning or error message. An understanding of the guidelines helps you build an effective fixture.

When is an XG-50 Fixture Necessary?

Some circuit boards might have devices that employ critical signals. For these devices, board noise could interfere with proper operation, and cause false failures. If your board includes a number of these devices, you should build an XG-50 fixture.

An XG-50 fixture improves the fidelity of all signals. The fixture improves grounding, which reduces the noise created by high-frequency digital signals. The XG-50 fixture provides additional ground paths, but eliminates most ground wires. This improves the signal fidelity of all digital signals.

To improve the signal fidelity of critical signals, the Fixture Generation Software operates with additional guidelines that take advantage of the XG-50 fixture's design. The software assigns critical signals to module interface pins that are near unblocked ground pins. An unblocked ground pin is a direct connection between the test system's ground and the fixture's ground plane. The software also tries to place ground probes near each critical signal probe.

An XG-50 fixture is necessary if your board test has:

- critical nodes
- high-speed digital tests (vector rates > 12.5 MHz)

When the `board` file is compiled, the board configuration file is checked for the correct pin cards depending on the fixture type specified. For the XG-50 fixture, the board configuration file must specify high-accuracy hybrid and channel cards.

Identifying Critical Signals

A critical signal is an edge-sensitive or pulse-sensitive input signal that is connected to a high-speed digital driver, but is not overdriven by the driver. This signal could cause the device to change state if an erroneous edge were detected. Most critical signals are sensitive clock lines and set/reset inputs on sequential devices or circuits. If such a signal inadvertently changes state, the state of the device or circuit also changes, and a false failure occurs during the device's test.

Problems with critical signals can occur during tests that have large numbers of high-slew-rate outputs changing states at the same time. This situation creates noise, which impairs the fidelity of critical signals: a common problem with digital functional tests.

The following steps help you identify critical signals on your board:

- Consider the logic families used on the board. If the board uses logic families with output edge slew rates greater than 400 V/usec (such as ACT devices), your board probably has some critical signals.

- Look for edge-sensitive and pulse-sensitive device inputs, which are typically clock inputs and set/reset inputs. If these inputs are connected to a driver (or clock resource) during testing and the driver is not used to overdrive an upstream output, the input node should be labeled as critical.
- If more than five percent of the nodes on a board are labeled as critical, you should reconsider which nodes truly are critical. Labeling less than five percent of the nodes as critical ensures the best resource and wiring selection by the Fixture Generation Software.

The board_xy compiler generates a warning message if more than five percent of the nodes are marked CRITICAL. Use the CRITICAL option sparingly to ensure the best resource and wiring selection by the Fixture Generation Software.

NOTE

You can specify which nodes have critical signals with Board Consultant. Refer to the Chapter 3, **Creating Board Information** for more information.

NOTE

For small boards, the board_xy compiler uses a fixed number to determine if too many nodes have been marked CRITICAL. This ensures that a reasonable number of critical nodes can be specified. If this number is exceeded, the board_xy compiler generates a warning message. For a large fixture, boards with fewer than 500 nodes can have up to 25 critical nodes specified. For a standard fixture, boards with fewer than 250 nodes can have up to 12 critical nodes specified.

Affects on the Fixture Generation Software

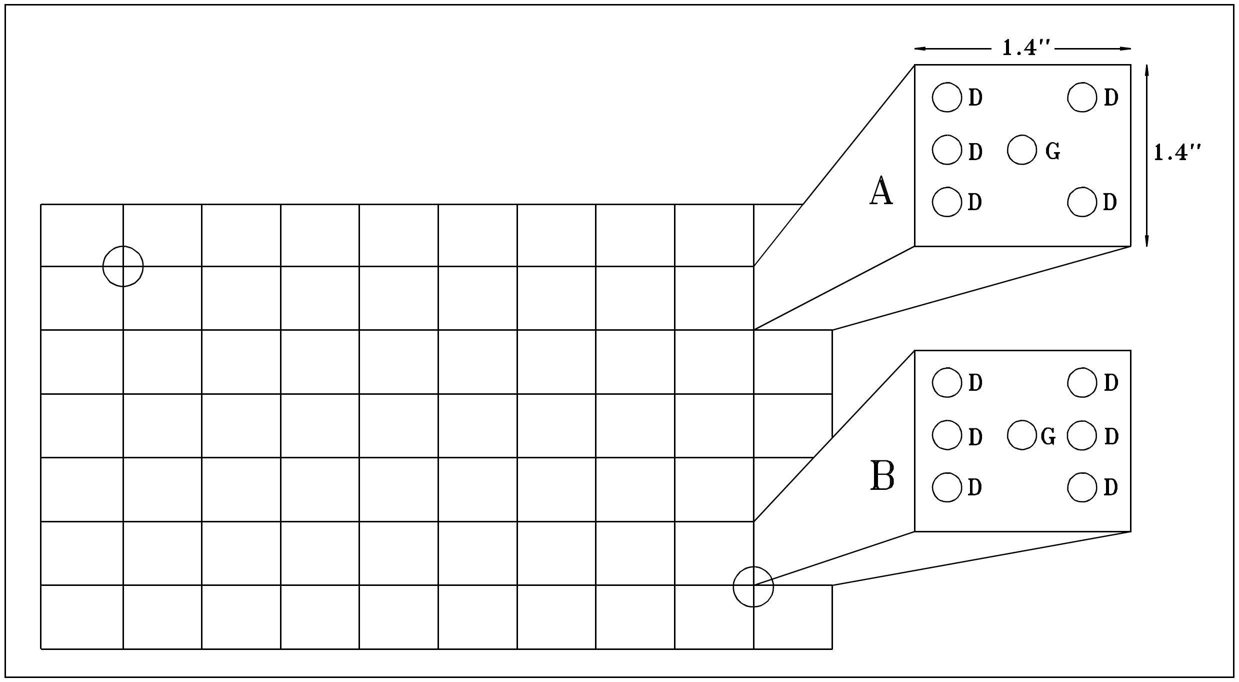
When an XG-50 fixture and critical nodes are specified, the Fixture Generation Software operates with additional guidelines. These guidelines ensure high fidelity for critical signals and help you build a high-quality fixture. This section discusses the additional guidelines that the Fixture Generation Software uses when you specify an XG-50 fixture.

Probe Select for the XG-50 Fixture

Probe Select, besides selecting probe locations, tries to assign an adequate number of ground probe locations distributed across the board for proper grounding. To disperse ground probes, Probe Select considers the board as if it were divided into 1.4-inch X 1.4-inch squares. Within each square, there should be no more than five digital probes for every ground probe.

Figure 4-11 shows an example of how Probe Select divides the board into a grid and examines each square of the grid for proper grounding. In this example, two of the squares are enlarged; ground probes are marked **G**, and digital probes are marked **D**. Probe Select considers square A, which has five digital probes per ground probe, to have acceptable grounding. Square B, however, has six digital probes per ground probe, which exceeds the desired limit.

Figure 4-11 Dispersing ground probes across the board



If any square contains too few ground probes, Probe Select issues a warning message. This message asks you to specify ground probe locations that are inside the square as MANDATORY. After doing this, run the Fixture Generation Software again. The message also includes:

- the percentage of the board that is deficient of ground probes
- the average ratio of digital probes per ground probe for all squares that exceed the desired limit

For each square deficient in ground probes, the X-Y coordinates of the digital probes are listed in the Miscellaneous section of the Details Report.

Probe Select avoids blocking module interface pins that are multiplexed to scarce or important resources. Because proper grounding is important for digital testing, the extended ground pins are considered important resources. Probe Select generates a warning if more than 15 extended ground pins are blocked on any pin card. The Details Report includes a `Blocked Pins` section. This section lists each card, by module number and card number, that has more than 15 ground pins blocked.

Probe Select generates another warning if three or more adjacent, extended ground pins are blocked on any pin card. The `Blocked Pins` section of the Details Report helps you identify which adjacent ground pins are blocked. The listing includes the pins' brc locations, and the reasons the pins are blocked (such as probe interference). Cards that have more than 15 ground pins blocked or three consecutive ground pins blocked are not used to provide resources for tests with critical signals.

After Probe Select has chosen all the probe locations, it checks each critical probe location for a nearby ground probe. The software looks for a ground probe location within one inch of the critical probe location. If it cannot find one, Probe Select issues a warning message. This message asks you to mark a ground probe location within one inch of the critical probe location as MANDATORY in the `board_xy` file. After doing this, run the Fixture Generation Software again.
X-Y coordinates of critical probes not placed within one inch of a ground probe location are listed in the Miscellaneous section of the Details Report.

MPA for the XG-50 Fixture

MPA has three guidelines it follows when assigning a critical node to a module interface pin. MPA tries to assign critical nodes to module interface pins according to the following guidelines:

- Assign a critical node to a pin within three pins of an unblocked switched ground pin.
- Maintain a buffer area around each pin assigned to a critical node; that is, signals involved in the same test as the critical node, cannot be assigned to pins within this buffer.

- Do not assign critical nodes or other connections in the test to cards that have an unacceptable number of extended ground pins blocked.

Besides the 38 extended ground pins on the high-accuracy pin cards, each pin card has six switched ground pins. The 38 extended ground pins on each high-accuracy pin card also operate as switched grounds. MPA must assign a critical node to a pin within three pins of an unblocked switched ground pin.

Figure 4-12 represents the row of module interface pins on a pin card. In this example, the critical pin is assigned to pin 36, which is three pins from ground pin 39.

If MPA cannot satisfy the first guideline, a fixture overflow occurs. A fixture overflow occurs when there are not enough resources available in the testhead. If this happens, MPA assumes that cards were added to your board configuration file and generates a warning message. In this case, the warning message lists the critical nodes that caused the fixture overflow and the cards that were added. The warning message indicates that because an XG-50 fixture was specified and the guideline could not be satisfied, the critical node assignment could not be made. MPA also generates an error message, and stops the test development process.

Similarly, if the second guideline cannot be satisfied, a fixture overflow occurs, and MPA generates warning and error messages. Starting at the switched ground pin associated with a critical node, and extending four pins beyond the critical signal pin, no other pins should be assigned. This provides a buffer between the critical signal and all other signals. The buffer is desirable only for tests that use the critical signal. If a test makes no connection to the critical signal pin, the pins within the buffer can be assigned for that test. Figure 4-13 on page 4-92 shows an example of the minimum buffer desired. In this example, the buffer extends from pin 33 to pin 38.

To resolve a fixture overflow caused by one of these two cases, you can take one of the following actions:

- Add the necessary cards to your testhead and board configuration file.
- Consider removing the CRITICAL attribute on the nodes causing the fixture overflow.

The third guideline prevents assignment of critical nodes or other connections in the test to cards with:

- more than 15 extended ground pins blocked, or
- three consecutive extended ground pins blocked

Probe Select detects this problem first and generates warnings. Information about the blocked pins is added to the Details Report. If this guideline cannot be satisfied, MPA does not use the affected cards to provide resources for tests with critical signals. Adding cards to the testhead helps this problem also.

Figure 4-12 Critical node assignment (within three pins of an unblocked ground pin)

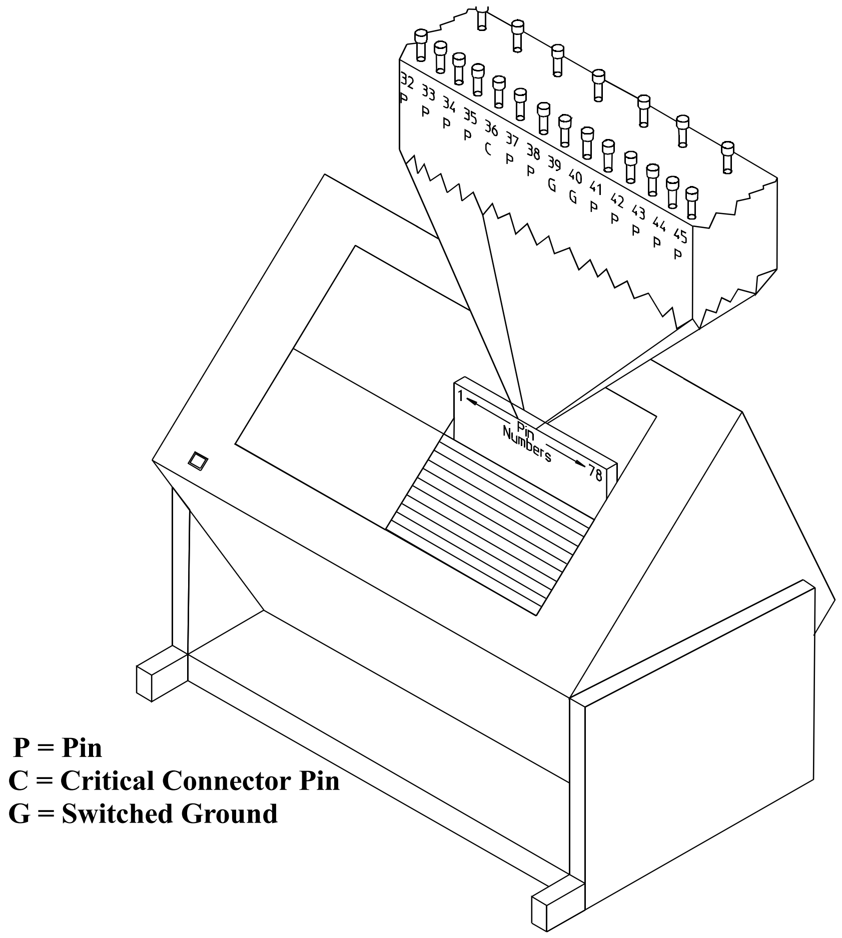
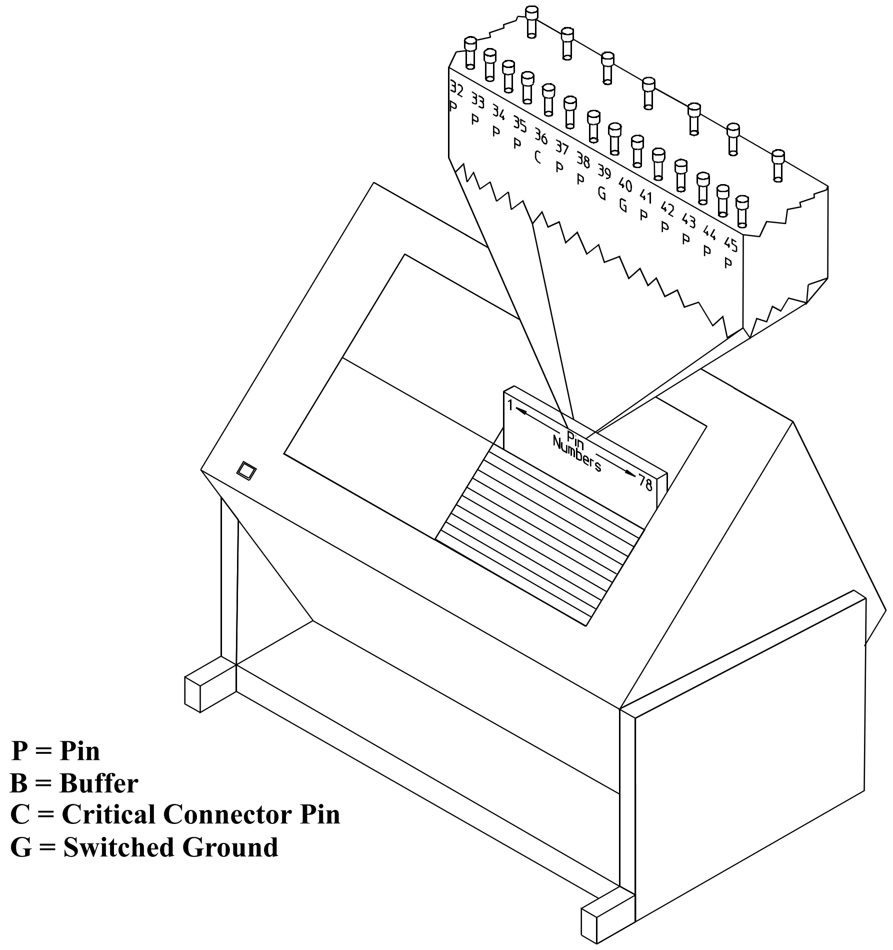


Figure 4-13 Critical node assignment (maintaining a buffer)



Evaluating Wire Lengths of Critical Nodes

MPA calculates the wire lengths for precision nodes, which include XG-50 critical nodes. If the wire length exceeds six inches for a precision node, MPA generates a warning message. This message instructs you to see the Details Report for a list of the precision nodes that were assigned long wire lengths.

Unless an XG-50 critical signal is extremely sensitive, long wire lengths for critical nodes do not adversely affect the signal for most tests. However, to be certain, you can use twisted-pair wires for your XG-50 critical signals. If you do this, you can solder both ends of the ground wire to the ground plane. Alternatively, you can wrap both ends of the ground wire to the nearest ground personality pins.

Fixture Tooling for the XG-50 Fixture

The program that generates fixture building reports/files is called Fixture Tooling. The fixture building reports contain information for drilling and assembling your fixture. One set of reports contains the X-Y drill information required to drill the various plates of the fixture, such as the probe plate. Fixture Tooling generates one additional drill file when you specify the XG-50 fixture. Called `drillgdp`, this file is required for drilling the internal ground plane.

Reading the Fixture Files and Reports

Table 4-33 summarizes the reports and files generated for each type of test fixture.

Note that if Fixture Tooling is run with the cartesian keyword, the X-Y data would be on a vertical Y axis and horizontal X axis; otherwise, the X-Y data would be on a vertical X axis and horizontal Y axis.

Table 4-33 Reports and files generated for test fixtures

Report / File	SimPlate	SimPlate Express Express Cassette	SimPlate XG-50 XG-50 Cassette
Wiring Report	Complete **	Top / Bottom *	Top / Bottom *
Inserts Report	Yes **	Top / Bottom *	Top / Bottom *
Trace Report	Complete **	Top / Bottom *	Top / Bottom *

Table 4-33 Reports and files generated for test fixtures (continued)

Report / File	SimPlate	SimPlate Express Express Cassette	SimPlate XG-50 XG-50 Cassette
Drill Files	None	Drill / Top / Sup *	Drill / Top / Sup / Gdp *
Summary File	Yes **	Yes	Yes
Details File	Yes	Yes	Yes
Difference Report	For changing an existing fixture.		
Handler Parameters handler_params	For the Express Cassette and XG-50 Cassette fixtures only.		

* These reports and files contain **top** information, only for the Express Cassette and XG-50 Cassette fixtures, only if **top probes allowed** is turned on.

**These reports are complete for the SimPlate fixtures even though there is less information than for the Express fixtures. They contain less information than report/file for the SimPlate Express fixtures. There is no brc for probes and no wire length for SimPlate fixtures.

The Fixture Drill Files

The drill files contain drill tool and X-Y coordinate information. The information is in a common format for numerically controlled machines, called Excellon Format 2.

There are slight differences in the drill files depending on the type of fixture being built. This is determined by the Fixture Part Number in the `fixture.o` file.

The possible drill files are included in Table 4-34.

Table 4-34 `drill` files

Type	Description
`drill`	Drilling information for the probe plate.
`drillsup`	Drilling information for the support plate.
`drilltop`	Drilling information for the top probe plate.
`drillgdp`	Drilling information for the ground plane in the XG-50 fixtures.

The type of fixture you are using determines which files are generated as shown in **Table 4-35**.

Table 4-35 `drill` files determined by fixture type

Fixture	Drill File Generated
SimPlate	No `drill` files are generated for this fixture.
SimPlate Express	`drill` and `drillsup`
Express Cassette	`drill`, `drilltop`, and `drillsup`
SimPlate XG-50	`drill`, `drillsup` and `drillgdp`
XG-50 Cassette	`drill`, `drilltop`, `drillsup`, and `drillgdp`

The Fixture Inserts Report

The Fixture Inserts Report contains information for inserting pins, receptacles, and probes. This report is slightly different for each of the three fixture types.

The Inserts Report for an SimPlate fixture lists the probe, node name, and probe location.

The Inserts Report for an SimPlate Express fixture gives information on the X-Y location of the hole, the brc coordinates of the pins, and the type of pin or probe to insert. This information is in addition to the probe, node name, and probe location. This is the only report that cross-references brc, node name, and device.pin.

The Fixture Trace Report

The Fixture Trace Report lists the wiring connections for each probe. The connections are listed in alpha-numeric order by node name. This report is a cross-reference from node name to probe location. You can use this report when troubleshooting the fixture wiring.

The Trace Report for the SimPlate and the SimPlate Express are the same except:

- The SimPlate Trace Report does not provide brc probe locations, wire length information, or **top** listings
- The SimPlate Express Trace Report does not provide **top** listings

The Fixture Wires Report

The Fixture Wires Report provides the information necessary to wire the fixture manually. The wire description specifies the wire gauge, color, and length. The connection includes the following source and destination terminator types, and their brc coordinates.

- Probe and ID
- Personality Pin
- Transfer Pin

Column labels and wire colors and PIN/PROBE designators are provided in local languages from a separate file.

The wire length is calculated to 0.5 inch (to centimeters in metric mode) so pre-cut and pre-stripped wrap wires can be used. The length reported is the insulated length. The recommended gauge is 28 for ordinary signal wires and power supply wires, and 30 for wires attached to 50-mil probes. The SimPlate fixture uses 26-gauge, twisted-pair wires.

The Wires Report is divided into four sections:

- Wires that connect brc's to brc's (pin-to-pin).
- Wires that connect brc's to probes (pin-to-probe).
- Wires connected in groups.
- Wires used in top-side probing; if top-side probes are used on the cassette fixtures.

The Fixture Difference Report

An example of a Difference Report is shown in **Example 4-1** below. Fixture Tooling appends the Difference Report to the Details Report. This report lists the wires that need to be added to or removed from the fixture. For each wire that needs to be changed, the report lists the length, gauge, and color of the wire. It also tells you where the wire is connected to and from.

Example 4-1

```
    ---------------------------------------------------------------------------
        3070 FIXTURE WIRE DIFFERENCE   Wed Nov 2, 1988 08:10:06 PM PAGE 1
        /boards/sample/fixture/details
        --------------------------------------------------------------------------------
        Length Gauge  Color          From      To
        ------|------|--------|--------------------|------------------------------------
        Remove
         3.0   28      blue      Pin (2 13.00 16.0)  Probe[2 16.80 15.6]
         3.0   28      blue      Pin (2 13.00 16.0)  Probe[2 16.80 14.9]
                                 . . .
        Add
         4.0   28   blue         Pin (2 13.00 19.0)  Probe[2 16.80 15.6]
         2.5   28   blue         Pin (2 13.00 19.0)  Probe[2 16.80 14.9]
                                 . . .
        Remove  15
        Add     6
```

The Fixture Summary Report

The Summary Report provides a summary of the information in the wires, inserts, trace, and drill reports. The information includes the numbers and sizes of holes drilled both completely and partially through; also the numbers of wires used in each color, gauge and length.

The Summary Report is stored in the `fixture/summary` file.

A Summary Report for an Express Cassette is shown in **Example 4-2**, below. The first section of the summary Report is provided by Board Placement. The second part of the report is added by Probe Select. Sections three and four are added by MPA and Fixture Tooling. The Summary Reports for the SimPlate and SimPlate Express fixtures do not include information about transfer pins and probes. The Summary Report for the SimPlate fixture does not provide wire-length information.

Example 4-2

```
-------------------------------------------------------------------------------
3070 Board Placement Summary Report Wed Nov 2, 1988 06:49:24 PM PAGE 1
/boards/sample/fixture/summary
-------------------------------------------------------------------------------
Inserted fixture part number "44201S" in the fixture object file.
The fixture part number was selected based on the fixture parameters
specified. If you wish to use an alternative fixture part number, please
list the fixture object and replace the fixture part number in the fixture
file with one of the other recognized part numbers described in the documentation.
Trying to find a placement for the board that is compatible
with the board handler.
The board "sample" was placed at X = 30198, Y = -77677
with a rotation of 0.000 degrees.
-------------------------------------------------------------------------------
3070 Probe Select         Wed Nov 2, 1988 06:34:23 PM PAGE 1
/boards/sample/fixture/summary
-------------------------------------------------------------------------------
     BOARD Sample PROBE REPORT FOR ALL NODES
     Nodes that are inaccessible or need more probes= 0
     Maximum probe force density (oz./sq. in.)   =  16
     Number of probes exceeding density threshold= 0
     Number of light spring force probes used    =  0
     Number of 50MIL probes used                 =  0
     Nodes marked for dual-stage probes that could
     not be assigned dual-stage probes           =  0
     Number of probes close to the edge of the board
     for a vacuum fixture                        =  0
     Number of probes close to the edge of the board
     for a mechanical fixture                    =  0
     Nodes added to the fixture                  =  0
     Nodes marked NO_PROBE or NO_ACCESS in board_xy=0
     Alternate probing locations discarded       =  0
     Pins blocked during this run of Probe Select= 0
     Blocked pins actually in the configuration =  0
     Blocked pins not in the configuration      =  0
--------------------------------------------------------------------------------
3070 MODULE PIN ASSIGNMENT     Wed Nov 2, 1988 07:10:01 PM PAGE 1
/boards/sample/fixture/summary
--------------------------------------------------------------------------------
  Module Pin Assignment was run at the novice experience level.
           ASSIGNMENT REPORT FOR ALL BOARDS
   SUMMARY STATISTICS
     For precision nodes:
       The average wire length is 1.707.
       The standard deviation is 1.484.
       * precision nodes include critical nodes, digital
        nodes, and analog nodes using external
        instrumentation or the time interval counter
     For all nodes:
       The assignment used 1102 personality pins for 980 nodes.
       The efficiency is 0.889.
--------------------------------------------------------------------------------
3070 FIXTURE TOOLING REPORT    Wed Nov 2, 1988 07:57:01 PM PAGE 1
/boards/sample/fixture/summary
--------------------------------------------------------------------------------
  Errors and Warnings...
--------------------------------------------------------------------------------
3070 FIXTURE TOOLING REPORT   Wed Nov 2, 1988 07:59:06 PM PAGE 2
/boards/sample/fixture/summary
--------------------------------------------------------------------------------
Fixture Type : Cassette
Fixture Size : Bank2
Top Probes Allowed : ON
Autofile : 4094
Metric Units : OFF
WireWrapping : Manual
--------------------------------------------------------------------------------
Latch:Long
-----
```

```
Count
-----
 10 Transfer Pins
 10 Transfer Probes
 408 Probes  Type: 100-mil Spring Force Max.: 8 oz
 546 Probes  Type: 100-mil Spring Force Max.: 4 oz
 125 Probes  Type: 75-mil Spring Force Max.: 8 oz
 26 Probes  Type: 50-mil Spring Force Max.: 4 oz
 18 Hybrid Cards
 94 Extra Pins
1248 Pins
  5 Tooling Pins
Count Color Gauge Length
------|-----|-----|--------
  3 Black  28  1.0 in
  6 Black  28  1.5 in
       . . .
  7 Black  28  3.5 in
---------------------------
  37 Black  28 154.5 in
       . . .
  1  Red  28  1.0 in
       . . .
  4  Red  28  2.0 in
---------------------------
  7  Red  28 55.5 in
  52  blue  28  1.0 in
  14  blue  28  1.5 in
       . . .
  2  blue  28  3.5 in
---------------------------
 175  blue  28 389.0 in
  1  Coax  30  8.5 in
  1  Coax  30 11.0 in
  1  Coax  30 13.0 in
---------------------------
```

```
   3  Coax  30  32.5 in
   1  Tw-pr 28  6.5 in
   1  Tw-pr 28  7.5 in
 ---------------------------
   2  Tw-pr 28  14.0 in
 ===========================
  1102       2497.5 in Total
```

The Fixture Details Report

An example of a Details Report is shown in **Example 4-3**. The Details Report contains all the information provided by the Summary Report with detailed explanations. When the Fixture Generation Software is run in incremental mode (for ECOs) the Details Reports also contains information about wiring that needs to be added or deleted from the existing fixture. This added information is the Difference Report, which is appended to the end of the Details Report. The Details Report is stored in the `fixture/details` file.

Example 4-3

```
-----------------------------------------------------------------------------
3070 Board Placement Details Report Wed Nov 2, 1990 06:29:24 PM PAGE 1
/boards/sample/fixture/details
-----------------------------------------------------------------------------
Inserted fixture part number "44201S" in the fixture object file.
The fixture part number was selected based on the fixture parameters
specified. If you wish to use an alternative fixture part number, please
list the fixture object and replace the fixture part number in the fixture
file with one of the other recognized part numbers described in the documentation.
Trying to find a placement for the board that is compatible
with the board handler.
The board "sample" was placed at X = 30198, Y = -77677
with a rotation of 0.000 degrees.
-----------------------------------------------------------------------------
3070 Probe Selection             Wed Nov 2, 1990 06:34:23 PM
/boards/sample/fixture/details
-----------------------------------------------------------------------------
                      ERRORS AND WARNINGS
The variable tooling pin at location 74500, 87650
conflicts with a testhead fixture location pin. You need
to install 5 spacers between the shoulder of the tooling pin bushing
and the probe plate. For a vacuum fixture, support plate
alignment pins are also required.
(WARNING PBS99)
Probe Select was not able to put enough probes on node "NODE_1".
Please see the Nodes Needing More Probes report in the details file
for further information.
(WARNING PBS77)
Personality pins providing scarce resources have been blocked.
Please see the Pins Blocked By Probe Select report in the details
file for further information.
(WARNING PBS100)
-----------------------------------------------------------------------------
```

```
3070 Probe Selection             Wed Nov 2, 1990 06:34:23 PM
/boards/sample/fixture/details
------------------------------------------------------------------------------
       NODES MARKED NO_PROBE OR NO_ACCESS IN BOARD_XY
  NODE_2
  NODE_3
  NODE_4
                NODES NEEDING MORE PROBES
Node: NODE_5
  Required probes: 1
  Selected probes: 0
  Probing locations not used:
    73800, 87000 [r77.2]
      Too close to tooling pin at 74500, 87650
      Clearance: 955
      Required clearance: 2850
    53200, 23010 [u20.9]
      Too close to probe P207 at 53200, 23210 [u20.10]
      Clearance: 200
      Required clearance: 490
Node: GND
  Required probes: 20
  Selected probes: 19
  No unused probing locations
------------------------------------------------------------------------------
3070 Probe Selection             Wed Nov 2, 1990 06:34:23 PM
/boards/sample/fixture/details
------------------------------------------------------------------------------
            PINS BLOCKED BY PROBE SELECT
Pin 11964 provides hybrid card resources
  Blocked by probes:
    P45  89500, 5500 [c55.2]Node: NODE_6
    P108 89550, 6100 [u33.7]Node: NODE_7
Pin 21862 provides GP relay (scarce)
Blocked by tooling pin at 5000, 17150
------------------------------------------------------------------------------
```

```
3070 Probe Selection             Wed Nov 2, 1990 06:34:23 PM
/boards/sample/fixture/details
------------------------------------------------------------------------------
                   MISCELLANEOUS
Light spring force probes:
  NODE_8 ( 27300,15500 [u19.12] )
------------------------------------------------------------------------------
3070 MODULE PIN ASSIGNMENT     Wed Nov 2, 1990 07:14:53 PM PAGE 1
/boards/sample/fixture/details
------------------------------------------------------------------------------
  Module Pin Assignment was run at the novice experience level.
           ASSIGNMENT REPORT FOR ALL BOARDS
  WARNING: The precision wires might cause problems due to their length
Wire between 21451 and node 1048 is 9.746 inches.
Node 1048 is used for precision analog wires.
                 . . .
Wire between 22376 and node 1241 is 7.302 inches.
  Node 1241 is used for analog and digital wires.
------------------------------------------------------------------------------
3070 MODULE PIN ASSIGNMENT     Wed Nov 2, 1990 07:13:53 PM PAGE 2
/boards/sample/fixture/details
------------------------------------------------------------------------------
Module Pin Assignment was run at the novice experience level.
           ASSIGNMENT REPORT FOR ALL BOARDS
   SUMMARY STATISTICS
     For precision nodes:
       The average wire length is 1.707.
       The standard deviation is 1.484.
       * precision nodes include critical nodes, digital
        nodes, and analog nodes using external
        instrumentation or the time interval counter
     For all nodes:
       The assignment used 1102 personality pins for 980 nodes.
       The efficiency is 0.889.
```

```
------------------------------------------------------------------------------
3070 FIXTURE TOOLING ERROR REPORT Wed Nov 2, 1990 07:57:03 PM PAGE 1
/boards/sample/fixture/details
------------------------------------------------------------------------------
  Errors and Warnings...
  Module #  Card #
  ----------------------
  Module 2  Card 2
  Module 2  Card 2
       . . .
  Module 3  Card 10
  Module 3  Card 11
              Critical Ground Wire Lengths
 Bank  # Row  #    Column # --wired to--Probe  # length
------------------------------------------------------------------------------
% Bank 2 Row  14   Column 20------------Probe  6 2.50 in.
 Bank  2 Row  15   Column 20------------Probe  6 1.50 in.
 Bank  2 Row  20   Column 60------------Probe  7 2.00 in.
% Bank 2 Row  14   Column 10------------Probe  11 3.00 in.
------------------------------------------------------------------------------
3070 FIXTURE WIRE DIFFERENCE   Wed Nov 2, 1990 08:10:06 PM PAGE 1
/boards/sample/fixture/details
------------------------------------------------------------------------------
 Length Gauge  Color        From                To
--------|------|-------|----------------------|---------------------------
Remove
 3.0  28   blue       Pin (2 13.00 16.0)  Probe    [2 16.80 15.6]
 3.0  28   blue       Pin (2 13.00 16.0)  Probe    [2 16.80 14.9]
                 . . .
Add
 4.0  28   blue       Pin (2 13.00 19.0)  Probe    [2 16.80 15.6]
 2.5  28   blue       Pin (2 13.00 19.0)  Probe    [2 16.80 14.9]
                 . . .
Remove 15
Add    6
```

The handler_params File

The `handler_params` file is a text file created by Fixture Tooling and stored under the board directory. This file is created only if you specified one of the cassette fixture types in Board Consultant. The `handler_params` file is used by the Express Fixturing System when testing your boards.

This file contains physical information about the board under test. The information includes the keywords in **Table 4-36**.

Table 4-36 Information from the `handler_params` file

Keyword	Description
Metric	Marks the beginning of a list of values in tenths of millimeters.
English	Marks the beginning of a list of values in mils.
Physical_length <length>	Maximum length of the board's driven edge.
Optical_length <length>	Length of the board's driven edge minus any notches.
Optical_tolerance <tolerance>	Tolerance of the board's optical length in mils or tenths of millimeters.
Width <width>	Maximum width of the board or the board carrier.
Placement <distance>	Distance from the fixture reference to the board edge.
Max_component_height <height>	Height of the tallest component on the board.
Thickness <thickness>	Thickness of the board.
Dual_stage_probes	Dual-stage (LONG) probes are present.
No_dual_stage_probes	Dual-stage (LONG) probes are not present.
Top_side_probes	Top-side probes are present.
No_top_side_probes	Top-side probes are not present.

Table 4-36 Information from the `handler_params` file (continued)

Keyword	Description
Right_to_left	Specifies the direction of the board through the board handler.
Left_to_right	Specifies the direction of the board through the board handler.

5 Building and Verifying the Fixture

In this chapter...

Objectives

When you finish reading this chapter, you should be able to:

- Use the fixture files and reports and the instructions in the *Building Board Test Fixtures* documentation to build the test fixture.

Prerequisites

Before you begin using this chapter, you should have already:

- Verified the wiring of the test fixture. The Agilent 3070 Series 3 provides some tools to help you accomplish this.

Required Tools and Materials

To accomplish the tasks in this chapter, you need:

- A 3070 Series 3 testhead to verify the test fixture.

Build and Verify the Test Fixture

To build the test fixture, use the fixture files and reports developed in Chapter 4.

A description of the verify features is given in **Table 5-1** on page 5-2. The `verify` syntax is described in **Table 5-3** on page 5-7. **Table 5-4** on page 5-10, **Table 5-5** on page 5-11, and **Table 5-6** on page 5-12 describes softkey shortcuts to be used with the `verify` statement.

1 **Build the test fixture.**

2 **Verify the wiring is correct.**

3 **Skip devices if necessary.**

4 **Verify the fixture.**

5 **Correct errors.**

6 **Verify the cassette fixture.**

7 **Verify multi-module test fixture.**

8 **Generate verification reports.**

9 **Check the wiring with the find pins feature.**

1 Build the test fixture.

2 Verify the wiring is correct.

After your test fixture has been built, you should verify that the wiring is correct. The 3070 Series 3 board test systems provide four features to help you verify the wiring of your fixture and to test connectivity from the testhead to the board under test. These features are detailed in **Table 5-1** on page 5-2.

Table 5-1 Verifying the wiring of your fixture

Feature	Description
verify	Use before production testing to ensure that your fixture is wired properly. This is explained in this chapter.
find pins	Use to help verify the fixture before and during production testing. It finds the connections to the node or device.pin that you touch with the guided-probe. Because this can be useful in fixture verification and production testing, it is explained in this chapter.

Table 5-1 Verifying the wiring of your fixture (continued)

Feature	Description
probe	Use to check connections to a device during production testing. It displays the pins of a device and indicates if there is a connection to each pin when you touch it with the guided-probe. This can be used interactively with the testplan to check contact to a device if it fails.
CHEK-POINT	Use during production testing to determine if the board under test is making contact to the fixture. This is a fast, coarse check, and should not be used to determine shorts and opens. You can use the board graphics with the `verify` and `find pins` features to help you quickly and easily locate the device or pin to be verified. Invoke the board graphics with the `board graphics` (BT-BASIC) statement.

a To verify the wiring of your fixture, ensure there is a guided-probe on your system, and it is listed in the `config` file.

b Execute the statements given in **Table 5-2** in the order listed.

Table 5-2 Statements to verify your test fixture

Statement	Description
testhead is 1	Specifies the testhead.
fixture lock	Activates the fixture locks on the testhead.
load board	Loads the `board.o`, `wirelist.o` and `shorts.o` files.

Table 5-2 Statements to verify your test fixture (continued)

Statement	Description
vacuum well statement	Maps fixture wells to vacuum ports. This is the same as the `vacuum well` statement in your testplan. If there is no `vacuum well` statement in your testplan, you do not need one here
fxon statement	Applies vacuum to the fixture. Use the appropriate statement for your fixture.
unpowered	Opens all the testhead relays.
printer is <filename>; echo	You can use the `printer is` statement to store the results of fixture verification in a file to be printed later. You also want to see the results on the screen. To save the results and see them on the screen, type: `printer is "file_name";echo` When finished, be sure to reset the printer by typing: `printer is *`
verify statement	Use the appropriate `verify` statement for the type of verify you want to perform.

c The verify feature lists the device or pin to be verified on the screen. You need to contact that device or pin with the guided probe.

d If **manual start** is on (set by softkey (F7), you need to either press the start softkey (F1) on the terminal, or step on the foot switch while contacting the device or pin with the guided probe.

e If **auto start** is on (set by softkey F7), you do not need to press the start softkey or step on the foot switch. The system checks the wiring from the point you are probing to the brc location of the interface pins.

After you press the start softkey or step on the foot switch, the audible feedback can be:

- three notes of the same pitch, indicating proper connections.
- two notes of different pitch, indicating an unexpected connection or open.

- four notes of different pitch, indicating an unexpected connection and an open.

  You can control the number of beeps for auto start mode with the `Verify.AutostartBeeps <#>` parameter of the `.hp3070` file; the default is four.

The verify feature applies 0.1 volt (to avoid turning on PN junctions) through a 10-ohm resistor. The threshold to determine a short or open is 10 ohms. Any measurement less than or equal to 10 ohms is considered a short. Any measurement over 10 ohms is considered an open.

If the wiring is correct, the test passes. If the wiring is incorrect (open to the correct interface pins or short to the incorrect interface pins), the test fails. Pass and failure information is listed on the screen and there is audible feedback unique to each kind of test result. The audible feedback reduces your dependence on the screen. Once you are familiar with the verification process and the feedback tones, you do not have to look at the screen after you probe a device.pin. This makes verification faster and easier.

Anytime you probe the fixture or board under test, the audible feedback can be:

- ■ a single tone, indicating that there is a connection to that point.
- ■ no tone, indicating that there is no connection to that point.

`Verify` can be either node-oriented or device.pin-oriented, depending on the string included with the `verify` statement. If you specify a node name, the statement is node-oriented; if you specify a device.pin, the statement is device.pin-oriented. The resulting report is always node-oriented.

The sequence for device.pin-oriented verify is alphabetical by device type. The system checks devices in this order:

a capacitors

b connectors

c diodes

d FETs

e functional

f fuses

g inductors

h jumpers

i library

j potentiometers

k resistors

l switches

m transistors

n zeners

3 Skip devices if necessary.

You can skip devices by using the `verify from` command. For instance, if you did not want to verify capacitors, you could type:

```
verify from "R1.1"
```

Devices are tested in reference designator order, C1, C2, ..., Cn, R1, R2, ..., Rn. The pins are tested in numerical order on each device. Using a device.pin-oriented verify process is easier and more thorough because of the audible pass/fail feedback. You can listen for the pass/fail tone and move on to the next device.pin without looking away from the board under test. You might have problems when the system skips a pin because it was not specified in Agilent Board Consultant or not used, or when you mis-probe a device. It is easy to be one pin off when you are probing a large SMT digital device.

4 Verify the fixture.

The `verify` statement allows the options described in **Table 5-3** on page 5-7.

There are three levels of fixture verification:

- Probe only the fixture probes from the top of the fixture.

  This method is the easiest but only verifies that the fixture wiring matches the `wirelist.o` file.

  Nodes with more than one probe present a problem here. Because there is no board on the fixture, multiple probes for one node are not shorted together. You see contact only to the probe you are touching with the guided-probe. The probes you are not touching look open to the system.

- Probe a blank board (no devices on the board) on the fixture with vacuum applied.

  This method is the best because it provides the most verification without added confusion. This not only ensures that the wiring matches the `wirelist` file, but also checks the accuracy of the `board` and `board_xy` files. If a probe is in the wrong X-Y location, an open or short occurs because the probe is contacting the board in the wrong place. If the topology of the board is incorrect, an inaccurate device.pin-to-node connection, open, or short results. To use this method on a vacuum fixture, you need to solder all holes in the blank board.

- Probe a loaded board on the fixture with vacuum applied.

  This is the same as using a blank board, but the results require interpretation. Any resistance less than the threshold of 10 ohms indicates a short. For example, the verify software would interpret two nodes separated by a 5-ohm resistor as a short.

Table 5-3 Syntax of the `verify` statement

verify	Specify one or a few device.pins, or nodes, or a combination. Note that node names that contain decimal points are treated as device.pins.
	`verify "U1-12"`
	`verify "U1.1"`
	`verify "U1-12","U1.1","R2-2"`

verify all	Test all device.pins on all nodes.
verify all nodes	Test one device.pin on each node.
verify from	Start the verification process from a specified device.pin. This is useful when you have been interrupted while verifying your fixture. You can come back later and continue where you left off. `verify from "U18.1"`
verify nodes from	As with the `verify from` statement, this allows you to start at a specified node. `verify nodes from "U1-12"`

5 Correct errors.

a If there is a wiring error inside the fixture, you need to change the wires involved.

b If there is an error in the `board` file, you have to correct it in Board Consultant.

Do not forget to run Develop Board Test after making your corrections. Fixture Tooling appends a Difference Report to the Details Report that tells you what wiring changes to make to your fixture.

Remember that if you are using a loaded board for verification, you need to investigate the errors. A 5-ohm resistor creates an error indication.

6 Verify the cassette fixture.

The Agilent Express Cassette and Agilent XG-50 Cassette fixtures present a special situation in verification. Because these fixtures are located inside the board handler during operation, they are not accessible with the guided-probe.

a Check the continuity of each wire with an ohmmeter.

Probe Select tries to minimize the number of probes in the top half, so this should not be a problem. You can use the fixture wires report to find the proper wiring connections. The fixture Trace Report tells you the node-to-transfer-pin connections.

b To verify the bottom half of a cassette fixture, place it on a testhead that does not have a board handler. Verify the same way as for an Express fixture (by contacting the fixture probes with the guided-probe).

If you do not use vacuum, you cannot use a blank or loaded board to verify the fixture. Use the fixture Trace Report to map the transfer pins to nodes.

c You can change your cassette fixture to a vacuum fixture to do a more thorough verification of the bottom half (using a blank board). You could use the fixture as a vacuum fixture for debugging the tests for devices probed from the bottom. Changing your Express Cassette to a vacuum fixture requires an Agilent 44200K Spare Parts Kit and Agilent 44200P variable diameter tooling pins.

NOTE

Refer to the instructions in the *Agilent Express Fixturing System* documentation for information about Building and Verifying the Cassette Fixture and Debugging Bottom-Plate Tests.

7 Verify multi-module test fixture.

For multi-module test fixtures, there can be G and L bus connections between the ASRU cards in each module.

Remember that the ASRU Cards are always located in slot one of each module. Refer to the testhead layout section of the *Quick Reference Guide* for more information.

The verify feature measures the resistance of each of these buses from the probe module to each other module used. If the measured resistance is greater than 10 ohms, the verify feature issues a warning to the print device:

```
Please check the wiring of the intermodule
G and L bus connections in the fixture.
```

a If you see this warning during the verify process, manually check the intermodule connections for the G and L buses.

Do not disassemble the fixture. Use the Fixture Wires Report to find which brc's should be connected.

b Use an ohmmeter to measure continuity from brc to brc on the personality pins protruding through the alignment plate.

For the G bus, any of the following brc's, in any module used by the board test, must be wired together in the fixture:

```
11154   12354 20125  21325
```

Note that for Bank 2, columns 25 and 27 can be used interchangeably; for Bank 1, columns 52 and 54 can be used interchangeably.

For the L bus, any of the following brc's, in any module used by the board test, must be wired together in the fixture:

```
11153  12353  20126  21326
```

Note that for Bank 2, columns 26 and 28 can be used interchangeably; for Bank 1, columns 51 and 53 can be used interchangeably.

8 Generate verification reports.

The format of the verification reports is determined by the type of verification-device.pin-oriented or node-oriented.

Reports for device.pin-oriented verify include:

- device.pin
- all expected brc's
- all untested brc's
- pass and failure information
- node name
- resistance of the measurement
- skipped devices
- backups

Reports for node-oriented verify include:

- all device.pins on node
- all expected brc's
- node name
- pass and failure information
- resistance of the measurement
- skipped nodes

When you check the ground node you see unexpected connections to the digital return grounds. These connections are not a problem and should be ignored. This only happens if you are using a blank or loaded board on the fixture.

9 Check the wiring with the find pins feature.

The find pins feature is useful for checking the wiring from any point on the fixture or board under test. You can use this feature to help debug the fixture wiring during fixture verification. You can also use it to help debug the testplan and to help isolate problems that arise during production testing.

a To use the find pins feature, type `find pins`.

b Hold the guided-probe on the point in question, and either press the start softkey (F1) or step on the foot switch.

The system finds all the connections to this point and lists them on the screen. After listing the results, the system instructs you to probe another point. This process continues until you press the stop softkey (F8).

a Use softkey (F6) to instruct the find pins feature to list connections to digital ground brc's.

Show all dig gnds causes the find pins feature to list every connection to a digital ground brc. **Show one dig gnd** causes the find pins feature to list only one connection to ground. Softkey F6 toggles between these two definitions each time you press it.

The threshold for determining shorts and opens is 10 ohms. Any resistance of 10 ohms or less is considered a connection. Any resistance greater than 10 ohms is considered an open.

Table 5-4 on page 5-10, **Table 5-5** on page 5-11, and **Table 5-6** on page 5-12 describes softkey shortcuts to be used with the `verify` statement.

Table 5-4 Softkey definitions for the `verify` statement

Softkey	Command	Definition
F1	**Start**	Check the connections to the point you are probing. Pressing this softkey is the same as stepping on the foot switch.
F2	**Skip Node**	Do not test the node currently called for on the screen and proceed to the next node in sequence.

Table 5-4 Softkey definitions for the `verify` statement (continued)

Softkey	Command	Definition
F3	**Re-try Node**	Display the node to be tested again. This is helpful if the display is long and part of it has rolled off the screen.
F6	**Show all dig gnds / Show one dig gnd**	Instructs the verify feature how to list digital ground brc's when it detects a short to ground. **Show all dig gnds** causes the verify feature to list every digital ground brc that it detects during a short to ground. **Show one dig gnd** causes the verify feature to list only one connection to ground. Pressing softkey F6 causes it to toggle between these two definitions.
F7	**Auto Start / Manual Start**	Instruct the system to start automatically or to wait for you to press the Start softkey or foot switch.
F8	**Quit Verify**	End the verify process.

Table 5-5 Softkey definitions for the `verify from` and `verify all` statements

Softkey	Command	Description
F1	**Start**	Check the connections to the point you are probing. Pressing this softkey is the same as stepping on the foot switch.
F2	**Skip Pin**	Do not test the pin currently called for on the screen and proceed to the next pin in sequence.
F3	**Re-try Pin**	Display the pin to be tested again. This is helpful if the display is long and part of it has rolled off the screen.

Table 5-5 Softkey definitions for the `verify from` and `verify all` statements (continued)

Softkey	Command	Description
F4	**Backup Pin**	Re-try the previous pin. Sometimes it is easy to probe the wrong pin on a large SMT digital device. If you see several consecutive failures and discover you have been one pin off, you can backup one pin each time you press this softkey.
F5	**Skip 1 Device**	Move ahead 1 device in the sequence each time you press this softkey.
F6	**Show all dig gnds / Show one dig gnd**	Instructs the verify feature how to list digital ground brc's when it detects a short to ground. **Show all dig gnds** causes the verify feature to list every digital ground brc that it detects during a short to ground. **Show one dig gnd** causes the verify feature to list only one connection to ground. Pressing softkey F6 causes it to toggle between these two definitions.
F7	**Auto Start / Manual Start**	Instruct the system to start automatically or to wait for you to press the Start softkey or foot switch.
F8	**Quit Verify**	End the verify process.

Table 5-6 Softkey definitions for the `verify from` and `verify all` statements

Softkey	Command	Description
F1	**Start**	Check the connections to the point you are probing. Pressing this softkey is the same as stepping on the foot switch.
F2	**Skip Node**	Do not test the node currently called for on the screen and proceed to the next node in sequence.

Table 5-6 Softkey definitions for the `verify from` and `verify all` statements (continued)

Softkey	Command	Description
F3	**Re-try Node**	Display the node to be tested again. This is helpful if the display is long and part of it has rolled off the screen.
F4	**Backup Node**	Re-try the previous node. Sometimes it is easy to probe the wrong pin on a large SMT digital device. If you see several consecutive failures and discover you have been one pin off, you can backup one node each time you press this softkey.
F5	**Skip 20 Nodes**	Move ahead 20 nodes in the sequence each time you press this softkey.
F6	**Show all dig gnds / Show one dig gnd**	Instructs the verify feature how to list digital ground brc's when it detects a short to ground. **Show all dig gnds** causes the verify feature to list every digital ground brc that it detects during a short to ground. **Show one dig gnd** causes the verify feature to list only one connection to ground. Pressing softkey F6 causes it to toggle between these two definitions.
F7	**Auto Start / Manual Start**	Instruct the system to start automatically or to wait for you to press the Start softkey or foot switch.
F8	**Quit Verify**	End the verify process.

6 Completing and Debugging Tests

In this chapter...

Objectives

When you finish reading this chapter, you should be able to:

- Complete any unfinished (setup-only) tests.
- Evaluate all tests. You can use the `ipg/summary` file to find information about the tests.
- Use Agilent PushButton Debug to help you debug and optimize tests.
- Use the Board Test Grader to evaluate how well the tests are working and to determine fault coverage of your test.

NOTE

Debugging analog tests is explained in Chapter 5, **Debugging Analog Tests** *in Test Methods: Analog.*

Debugging digital tests is explained in Chapter 6, **Debugging Digital Tests** in *Test Methods: Digital*.

See **Test Development Tools** for information on PushButton Debug and on the Board Test Grader.

Prerequisites

Before you begin using this chapter, you should have already:

- Built your test fixture.

Required Tools and Materials

To accomplish the tasks in this chapter, you need:

- Agilent 3070 Family testhead.

Complete, Evaluate, and Debug Tests

You need at least one known-good board available for this step. Using several boards to evaluate and debug the tests helps ensure success during production testing.

1 **Evaluate and debug CHEK-POINT.**

2 **Evaluate and debug the pre-shorts tests.**

3 **Evaluate and debug the shorts and opens test.**

4 **Evaluate and debug analog in-circuit tests.**

5 **Turn on the DUT power supplies.**

6 **Complete, evaluate, and debug digital tests.**

7 **Complete digital functional tests.**

8 **Debug digital tests.**

9 **Complete and debug analog functional and mixed tests.**

10 **Run the board test grader.**

1 Evaluate and debug CHEK-POINT.

Agilent IPG generates a file called `pins`. This file is a subset of the `shorts` file. The `pins` file is used by CHEK-POINT to verify that the board under test is contacting the fixture properly during production testing. CHEK-POINT applies a voltage to each node while grounding all other nodes and looks for leakage current. If it finds leakage current, the test passes; if it does not find leakage current, the test fails. CHEK-POINT is turned on with the `Usage Flags` in the testplan. You can enable CHEK-POINT to run for each board, or you can set it to run only on boards that have a failure.

a Execute CHEK-POINT and evaluate the results.

b Add a `report limit <#>` statement as the first line in the `pins` file to instruct CHEK-POINT to stop listing failures after the specified number.

This prevents a very long report when the entire board is not contacting the fixture.

c If CHEK-POINT finds any open nodes, investigate the problem.

Either the board is not contacting the fixture or there is no leakage path from that node due to circuit topology. If the fixture and board under test are making contact, and there is still no leakage current, you can remove the node from the `pins` file.

2 Evaluate and debug the pre-shorts tests.

a Evaluate the analog in-circuit tests that are executed before shorts and opens testing.

Pre-shorts tests usually include tests for devices such as potentiometers, jumpers, and switch settings. These tests are executed first because the position of potentiometers, jumpers, and switches affect shorts and opens testing.

NOTE

You can use the procedures for analog in-circuit debug as described in Chapter 5, **Debugging Analog Tests** *Test Methods: Analog*.

3 Evaluate and debug the shorts and opens test.

IPG uses the board topology information, `board.o`, a threshold resistance, and a settling delay to determine which nodes are considered as shorted (calculated impedance less than the specified threshold). IPG produces a `shorts` file that is used for shorts and opens testing. Shorts and opens testing checks for opens between nodes that are known to be shorted, and for shorts between all other nodes.

NOTE

Shorts and opens testing, including debug information, is included in Chapter 1, **Shorts & Opens Testing** in *Test Methods: Analog*.

a Use a known-good PC board to evaluate the shorts and opens testing.

Problems you might find with shorts and opens testing include:

- Shorts introduced by internal paths through devices on the board.
- Intermittent tests due to measured resistance very close to the specified threshold.
- Reactive devices still in their active region due to an insufficient settling delay.

b To debug the shorts and opens testing, you must edit the `shorts` file.

4 Evaluate and debug analog in-circuit tests.

a Use the Analog Debug utility and Histograms to evaluate and debug the analog in-circuit tests.

You can find information about the tests that IPG wrote in these files:

- `ipg/summary`
- `summary`

The debug utility works interactively with the analog test. You can use it to change bus connections and measurement parameters. The histograms show you the test results after each change you make.

NOTE

Refer to Chapter 1, **Agilent Pushbutton Q-STATS** in *Information Management* for information about the histograms.

b Update the `testorder` file and the testplan, if needed.

5 Turn on the DUT power supplies.

a Start with a low current limit and gradually increase it until the voltage comes up to the set level. Note the current setting and increase it by approximately thirty percent.

If the voltage does not increase as you increase the current limit, there is a problem with the fixture or the board under test.

b Optimize DUT power supplies

The `optimize` keyword instructs the system to optimize test throughput by turning on the power supplies simultaneously and by combining wait times. The testplan generator automatically includes the `optimize` keyword with the `sps` statements in the testplan.

The following paragraphs and examples explain how optimized power supplies are turned on. Note that the order in which optimized power supplies are turned on, is not controlled by the order of the `sps` statements in the testplan. If the timing or order of the power supplies is critical for your board test, you can turn off optimizing for some or all the power supplies by removing the `optimize` keyword from the appropriate `sps` statements.

Optimized power supplies increase test throughput two ways:

- The wait times are combined for a series of adjacent `sps` statements. All supplies in the series of `sps` statements are turned on simultaneously, then a single wait occurs. The wait is equal to the longest specified wait in the `sps` statements; if no wait is specified, the system uses 30 mSec. This happens every time you run the testplan, including the first run.

- Test throughput is further increased for the Agilent 6621 and Agilent 6624 power supplies by using the `store/recall` feature. This feature allows the settings for all outputs to be stored and later recalled with a single command. Recalling the stored states is much faster than programming all outputs individually. The settings are stored the first time you run the testplan (`nrun=1`); the settings are recalled on the second and subsequent runs (`nrun>1`).

The following examples show how optimized power supplies are turned on:

- Turn on supplies 1, 2, and 3 simultaneously with one 30 msec. wait.

```
sps 1, 4; optimize
sps 2, 5; optimize
sps 3, 10; optimize
```

- Turn on supplies 1, 2, and 3 simultaneously and wait 50 msec.

```
sps 1, 4; optimize
sps 2, 5, wa 50m; optimize
sps 3, 10; optimize
```

- Turn on supplies 1 and 2 simultaneously and wait 30 msec., then turn on supply 3 and wait 50 msec., then turn on supplies 4 and 5 simultaneously and wait 30 msec.

```
sps 1, 4; optimize
sps 2, 5; optimize
sps 3, 5, wa 50m
sps 4, 20; optimize
sps 5, 20; optimize
```

c If you have to connect DUT power supplies in parallel to achieve the desired current, check that the fixture wiring for the parallel DUT power supplies is connected to all module interface pins that are listed in the fixture wiring report.

Parallel DUT power supplies should be programmed differently in the `Setup_Power_Supplies` subroutine of the testplan. This is done automatically for you if you entered the parallel supplies in Board Consultant. One of the supplies that are connected in parallel should be programmed to the needed voltage value, the others should be programmed to 1% higher than the needed voltage. This places the first supply in constant voltage mode to control the other supplies which are in constant current mode. Specify the current for each supply as shown in **Table 6-1** on page 6-6.

You can verify that the supplies are programmed properly after they are turned on by observing the front panels. With power applied to the board under test, one of the parallel supplies should indicate constant voltage with the CV indicator lamp on the front panel; the others should indicate constant current with the CC lamp on the front panel.

Table 6-1 Specifying current for parallel DUT power supplies

Supply	Current	Example
For the supply set to the lowest voltage value	Use half the maximum current capability of the supply plus half the expected current swing of the board under test; we recommend that you increase the value to the next integer.	The board under test requires 10 to 12 amps (swing of two amps) and you are using three supplies capable of five amps each: Use half it's capability (5 / 2 or 2.5 amp) plus half the expected swing (2 / 2 or 1 amp) = 3.5 amps; round up to 4 amps.
For the other supplies	Use the maximum requirement of the board under test minus half the capability of the first supply divided by the number of other supplies.	The board under test requires 10 to 12 amps (swing of two amps) and you are using three supplies capable of five amps each: Use the total requirement (12 amps) minus half the first supply's capability (5 / 2 or 2.5 amps) = 9.5 amps divided by the number of other supplies (9.5 / 2) = 4.8 amps each.

d If you have to connect DUT power supplies in series to achieve the desired voltage, change the fixture wiring for DUT power supplies.

The left side of **Figure 6-1** on page 6-8 represents what the fixture wiring should look like if you entered the supplies in Board Consultant.

The right side of **Figure 6-1** on page 6-8 represents the changes you need to perform on the wiring.

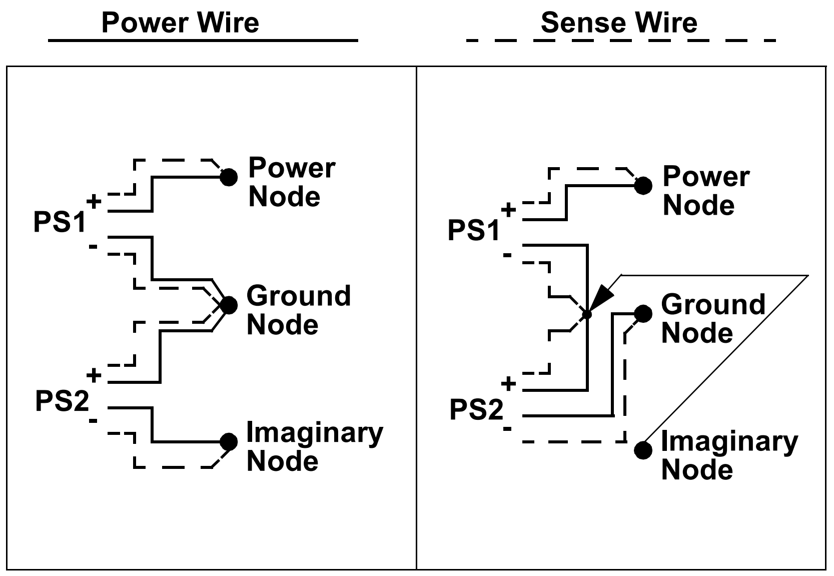
If you entered the supplies in Board Consultant, refer to **Figure 6-1** on page 6-8 and follow these steps:

- The low side of one supply and the high side of the other supply are connected to the ground node; remove this connection from the ground node but maintain the connection between the two supplies.
- The low side of one supply is connect to an imaginary node; move this connection from the imaginary node to the ground node.

If you did not enter the supplies in Board Consultant, see the right side of **Figure 6-1** on page 6-8 and follow these steps:

- Connect the low side of one supply to ground.
- Connect the high side of the same supply to the low side of the next supply (this connection should not include any board node).
- Connect the high side of the second supply to the power node.
- DUT power supplies connected in series should be programmed differently in the `Setup_Power_Supplies` subroutine of the testplan. Program each of the supplies to half the total desired voltage. Program one of the supplies to the desired current limit; program the other supply to 1% higher than the desired current limit.

Figure 6-1 DUT power supplies connections when wired in series



6 Complete, evaluate, and debug digital tests.

The digital tests include in-circuit and functional. The only difference between digital in-circuit and digital functional testing is that digital functional testing allows backtracing of internal nodes.

a Generate test patterns, either manually or downloaded from a CAE system, and finish writing the tests.

Test patterns generated on CAE systems and simulators must be changed to Pattern Capture Format (PCF).

To load the PCF file into the workspace in digital mode so it is syntax-checked, type the following and press **Return**:

```
load digital "file_id"
```

You can merge these files with other VCL source files before compiling them, or you can merge them at compile time with the `include` (VCL) statement.

b Compile the new or modified tests.

c Evaluate all digital tests and debug the tests that are marginal or failing.

NOTE

Please refer to Chapter 6, **Debugging Digital Tests** in *Test Methods: Digital* for a description of the digital debug process.

d Update the `testorder` file and the testplan, if needed.

e To compile a digital test, type

```
compile "<filename or pathname>"
```

on the BT-BASIC command line and press Return.

You can also use the optional `list` and `debug` keywords.

f You must use `debug` if you intend to use the Debug software to look at the expected input and output states for the test:

```
compile "<filename>"; debug, list
```

The files for library setup-only tests are not affected by this compilation and are not used again in this procedure unless you re-run IPG.

7 Complete digital functional tests.

You need to finish writing the digital functional tests generated as setup-only tests. You must perform the tasks described for digital in-circuit tests also for digital functional tests. In addition, you need to make sure the backtrace information is complete and correct and generate the internal node state data required for debugging and backtracing.

a Complete the tests as described in task 6 for digital in-circuit tests.

b Edit and compile the `backtrace` files.

If you are not using IPG Test Consultant, manually compile the backtrace files. This is an example:

```
compile "functional/ram/backtrace";list
```

The compiler stores the object code in the backtrace.o files.

These files contain the `backtrace trees`—connectivity and backtrace information used by the backtracer when a functional test fails.

c Generate internal node state data.

The internal node states are part of the data required for backtracing if a failure occurs during execution of a functional test. In any functional test, the internal nodes are those nodes in the circuit being tested but are not driven or received during test.

IPG identifies the internal nodes from the trace information in the library test. Module Pin Assignment assigns receiver pins to the internal nodes for backtracing. If backtracing is required, node state data must be supplied for every internal node that is to be backtraced.

Node state data can be downloaded from another system or learned from a known-good board using autolearn.

If you intend to learn the node states, you should wait until you have debugged the test. Although it is possible to write the internal node state data manually, this is not recommended.

d Compile the `states` files.

The files containing the node state data are called `states` files, and are located under the directories for the functional tests; for example, `functional/rom/states`. If you are not using IPG Test Consultant, compile these files with the `list` option, if required:

```
compile "functional/<test name>/states"; list
```

The compiler places the node state object code for each functional test in the `states` file, generated by IPG.

8 Debug digital tests.

a Examine the `ipg/summary` and `summary` files to see the status of the digital tests.

These files tell you if IPG commented out any vectors or units due to topology. Sometimes you can modify the test vectors to test more of the device pins.

b To debug digital tests, use the Debug software.

Use it interactively to observe the expected and actual states on the input and output pins of a device. Debug also lets you change certain test parameters and rerun a device test without re-compiling it.

Debug lets you compile any digital test (in-circuit or functional). For functional tests, Debug places the device's expected pin states in the `states` file so they can be displayed. The `states` file can be learned with autolearn.

NOTE

AUTOLEARN is not supported in MS Windows systems.

c For digital in-circuit tests, the Debug software lets you look at the expected and actual states of the device input and output pins. You can also (temporarily) change many of the test parameters—such as vector timing—and rerun the test without having to re-compile it. Save the changes in a file if you want to incorporate them permanently into your test; otherwise, the changes are lost when you exit the Debug software.

d For digital functional tests, the Debug software also lets you examine the circuit's internal nodes. If the states of the internal nodes are not yet known, you can use the Autolearn feature to learn them.

e Once a functional test that does not already have internal node state data is running properly, you can use Autolearn to learn that data. Autolearn stores the data directly into the `states` file.

f You must also learn the states for any nodes that need to be manually probed. These nodes should have been listed in the `board_xy`, file so the Fixture Generation Software would not assign interface pins to them. IPG generates an `sr, manual` test statement (`sr` means stored response) for each of these nodes and places it in the backtrace file.

9 Complete and debug analog functional and mixed tests.

During this task you need to complete the analog functional and mixed tests that are setup-only tests. You also need to evaluate all analog functional and mixed tests and debug the tests that are marginal or failing. You can use Pushbutton Debug, just as for analog in-circuit tests.

NOTE

For information about analog functional and mixed testing, please refer to Chapter 4, **Analog Functional and Mixed Testing** in *Test Methods: Analog*.

10 Run the board test grader.

We recommend that you run the Board Test Grader, as described in Chapter 9, **Board Test Grader, Test Coverage, & Coverage Analyst** in *Test Development Tools*, to evaluate the quality of the device tests and the fault coverage.

Summary

Any tests that were originally written as setup-only have been completed. All device tests have been evaluated to decide which were marginal or failing. And finally, the appropriate tests have been debugged.

File Structure for Step 5

The only additional files created at this step are the debug versions of the test source files:

```
digital/<device_name>.d
```

7 Production

In this chapter...

Objectives

When you finish reading this chapter, you should be able to:

- Release the test and fixture to production.
- Support production testing.
- Implement changes due to Engineering Change Orders (ECOs).

Prerequisites

Before you begin using this chapter, you should have already:

- Completed any unfinished (setup-only) tests.
- Evaluated all tests.
- Used the Board Test Grader to evaluate how well the tests are working and to determine fault coverage of your test.

Required Tools and Materials

To accomplish the tasks in this chapter, you need:

- Agilent 3070 Family testhead.

Release to Production

1 **Clean up, archive, and move the board directory.**

2 **Set optional test features.**

3 **Implement the board graphics viewer.**

4 **Turn off object checking.**

1 Clean up, archive, and move the board directory.

a Use BT-BASIC or the Board Management function of Agilent IPG Test Consultant to remove unnecessary files and directories from your board directory.

b Use the Board Management function of IPG Test Consultant to archive the board directory by copying it to tape.

c Use BT-BASIC or the Board Management function of IPG Test Consultant to move your board directory to the `$AGILENT3070_ROOT/boards` directory.

NOTE

With Agilent 3070 software revision 3070 04.00pa, an environment variable was created so that files can be easily transferred between UNIX® and MS Windows® controllers, which have different file systems. The environment variable, `$AGILENT3070_ROOT`, replaces the upper path names on both systems. For example, the `$AGILENT3070_ROOT` factory default value is /var/hp3070. In this document, only path names using the environment variable are used. If you must use actual path names, refer to older versions of the documentation. Please see **The Root Directory Environment Variable** in *Administering Agilent 3070 UNIX Systems* for further information.

2 Set optional test features.

There are optional test features you can set for production testing. These features include:

- Agilent PushButton Q-STATS
- Agilent PR PLUS
- CHEK-POINT
- datalogging
- boundary-scan

To set these features, edit the `Usage Flags and Other Parameters` in testplan. Load the testplan and type:

```
find "sub Set_Custom_Options"
```

The options that you can set are shown in **Example 7-1**.

a To turn on datalogging, set the `QSTATS_Mode` flag to `No_Histo` or `Histo` and set the `Analog_Sample_Rate` parameter to a non-zero value.

b To set CHEK-POINT to be executed every time, set the `Chek_Point_Mode` flag to `Pretest`; to execute CHEK-POINT on failed devices only, set the flag to `Failures`.

c If the test includes boundary-scan tests, set the `Using_BScan` flag to `True`.

d Be sure to re-save the testplan.

NOTE

For a description of Pushbutton Q-STATS and PR Plus see Chapter 1, **Agilent Pushbutton Q-STATS** in *Information Management*.

Example 7-1

```
!  Usage flags
     QSTATS_Mode               = Off       ! Can be: Off, No_Histo, or Histo.
     Chek_Point_Mode           = Failures  ! Can be: Off, Pretest, or Failures.
     Using_BScan               = False     ! Can be True or False.
     Using_PRPlus              = False     ! Paperless Repair Plus.
     Serializing               = False     ! Set True if Using_PRPlus.
     Using_Buffered_Reporting= True        ! Report failures during board handling.
   !  Other parameters
     Report_Printer$     = "/dev/rpr1"  ! Final report destination.
     Testrev$            = "RevA"
     Analog_Sample_Rate  = 0.1          ! Meaningful when Using_Histograms
     Serial_Length       = 28           ! Board Id Length (0 = no checking)
```

3 Implement the board graphics viewer.

You can implement the Board Graphics Viewer to help the board test operator identify parts on the PC board. Be sure to re-save the testplan.

4 Turn off object checking.

For faster test throughput, you can add the `object checking` statement to the testplan to turn object checking off. This instructs the system to check the timestamp of test object files only the first time each test is run, instead of every time each test is run.

Support Production Testing

The 3070 Series 3 provides tools to help you during production testing.

1 Use statistical quality control tools.

2 Check the fixture or board wiring.

3 Check the connections to all used pins of a device.

4 Ensure the fixture is making contact with the testhead.

5 Add custom ease-of-use functions for the operator.

6 Enter a temporary programming session.

7 Maintain the test fixture.

1 Use statistical quality control tools.

The 3070 Series 3 provides SQC tools to monitor and improve your manufacturing processes.

NOTE

For information about Data Logging and Pushbutton Q-STATS refer to Chapter 1, **Agilent Pushbutton Q-STATS** and Chapter 2, **Datalogging** in *Information Management*.

Use the tools shown in **Table 7-1** during production testing and troubleshooting

.

Table 7-1 Tools used during production testing and troubleshooting

Tool	Description
find pins	Use to help verify the fixture before and during production testing. It finds the connections to the node or device.pin that you touch with the guided probe. Since this can be useful in fixture verification and production testing, it is explained in this chapter and in *Chapter 7,* **Production.**
probe	Use to check connections to a device during production testing. It displays the pins of a device and indicates if there is a connection to each pin when you touch it with the guided probe. This can be used interactively with the testplan to check contact to a device if it fails.
CHEK-POINT	Use during production testing to determine if the board under test is making contact with the fixture. This is a fast, coarse check, and should not be used to determine shorts and opens.

2 Check the fixture or board wiring.

The `find pins` feature is useful for checking the wiring from any point on the fixture or board under test. For more information, see **Check the wiring with the find pins feature.** on page 5-10.

The threshold for determining shorts and opens is 10 ohms. Any resistance of 10 ohms or less is considered a connection. Any resistance greater than 10 ohms is considered an open.

Example 7-2

```
Position the Probe, then "START".
Found (21954), 0 ohms.
Node: "R1-1"
          R1.1
          C4.2
          U1.5
Found (22063), 5 ohms.
Node: "R1-2"
          R1.2
          U2.9
Position the Probe, then "START".
Ending Find Pins function.
```

In **Example 7-2**, above, case R1 is a 5 ohm resistor. Since 5 ohms is less than the threshold, find pins found the nodes on both pins. After finding the connections to the point probed, the user pressed the stop softkey (F8) and find pins halted.

3 Check the connections to all used pins of a device.

a Add the following line to the testplan before the device or devices you wish to check:

```
save failures on
```

b Add the following line to the testplan after the device or devices you wish to check:

```
save failures off
```

c Add the following line at the end of the subroutine:

```
probe failures
```

d Use the `board number is` statement to control failure probing of multiple boards on one fixture. **Example 7-3** shows how to probe the failing devices on each failing board of a two-board fixture, and send the logging and reporting information to two different files.

Example 7-3

```
buffered reporting on
save failures on
log is *
log level is failures
report is *
report level is report
Test:
  board number is 1
  test "analog/r1_A"
  test "analog/r2_A"
       .
       .
       .
  board number is 2
  test "analog/r1_B"
  test "analog/r2_B"
       .
       .
       .
for I = 1 to 2
  board number is I
  if boardfailed then
    probe failures
    log out "LOGFILE_B" & val$(I); append, echo
    report out "REPFILE_B" & val$(I); append, echo
  end if
next I
board number is *
goto Test
```

The `probe` function checks the connections to all used pins of a device. It checks devices that have up to 650 pins. The `probe` function displays the pins of the device on the screen. The display indicates the pins that are used with an * (asterisk), and the pins that are not used with a . (period). As you probe each pin with the guided probe, the * changes to a `p`. After you remove the guided probe the `p` changes to a `T`. If no contact is detected to a used pin, the * remains on the screen. **Example 7-4** demonstrates this. This does not require the use of the start softkey or the foot switch. You can probe the pins in any order and repeat any that you want. Probe the devices carefully so that you do not damage any device leads, especially on SMT devices.

The `probe` function applies 0.1 volt through a 10 ohm resistor to the point you touch with the guided probe. It uses a 10 ohm threshold to determine a short.

To use the probe function, type `probe "<device>"` such as:

```
probe "U14"
```

The probe function re-defines the softkeys as described in **Table 7-2**.

Example 7-4 Probe report of a 28-pin device with 5 unused pins.

Before probing the screen looks like this:

```
Probing Device: U14
Pins>     1 1111111112 2222222223 3333333334 4444444445 T Tested
     1234567890 1234567890 1234567890 1234567890 1234567890 p Probe
     ------------------------------------------------------ * Untested
  0 + ********** ****.***** **....**.. .......... .......... . Unused
```

As each pin is probed the "*" changes to a "p":

```
Probing Device: U14.7, "CLOCK"
Pins>     1 1111111112 2222222223 3333333334 4444444445 T Tested
     1234567890 1234567890 1234567890 1234567890 1234567890 p Probe
     ------------------------------------------------------ * Untested
  0 + TTTTTTp*** ****.***** **....**.. .......... .......... . Unused
```

After each pin is probed the `p` changes to `T`. No connection was found for pin 10:

```
Probing Device: U14.28, "VCC"
Pins>     1 1111111112 2222222223 3333333334 4444444445 T Tested
     1234567890 1234567890 1234567890 1234567890 1234567890 p Probe
     ------------------------------------------------------ * Untested
  0 + TTTTTTTTT* TTTT.TTTTT TT....TT.. .......... .......... . Unused
```

Table 7-2 Softkeys redefined by the `probe` function

Softkey	Command	Description
F2	**Abort Device**	Stop the probe function for the displayed device.
F3	**Re-try Device**	Re-probe the displayed device.
F4	**Next Device**	Move to the next failed device. This softkey is used only with the `probe failures` statement.
F5	**Hardcopy Printer**	Copy the screen to the `printer is` device.
F6	**Report Untested**	List the untested pins on the `report is` device.
F8	**Quit Probe**	End the probe function.

Many times, more than one pin of a device is connected to the same node. When you probe any of these pins a **p** appears for all pins that are on that node. The report lists the lowest numbered pin. Note that since the threshold is 10 ohms, any impedance less than 10 ohms looks like a connection.

For devices with more than 50 pins the display is repeated. The 0 + in the last row increases by 50 each time. This indicates how much to add to the pin number. For the first display, add 0 to the pin number. In a second display you would add 50 to the pin number.

In **Example 7-5**, the first row displays pins 1 through 50, and the second row displays pins 51 through 100 (50 plus the pin number). Since this is a 75 pin device the display indicates that pins 76 through 100 are not used. In this example, pins 1 through 18 have been tested with no contact being found for pin 15. Pin 19 is being probed, and the rest have not been tested yet.

Example 7-5 A probe report for a 75 pin device

```
Probing Device: U14.5, "CLOCK"
    Pins>     1 1111111112 2222222223 3333333334 4444444445 T Tested
         1234567890 1234567890 1234567890 1234567890 1234567890 p Probe
          ------------------------------------------------------ * Untested
       0 + TTTTTTTTTT TTTT.TTTp* **....**.. ********** *****.**** . Unused
       50 + ********** *....***** **..*..... .......... ..........
```

The `probe` function can be used during production testing to check connection to any failing device. If a device was not soldered well or if the board under test has marginal probing areas, such as vias on a dense SMT device, use the `probe` function to check connections to this device, if it fails. This determines if it is truly a bad part or if one or more pins did not make contact during testing.

4 Ensure the fixture is making contact with the testhead.

a Invoke CHEK-POINT by typing:

```
test "pins"
```

There are three options to the CHEK-POINT feature. The three options are:

- Do not execute CHEK-POINT.
- Execute CHEK-POINT on every board.
- Execute CHEK-POINT only on boards that have a device failure.

b Turn CHEK-POINT on and off with the `Usage Flags` in testplan.

During production testing, if CHEK-POINT finds an open it asks the operator to press **YES** to try again or **NO** to stop. After a second try, CHECK-POINT asks the operator to press **YES** to cycle the vacuum and try again, or press **NO** to stop. After a third try, CHEK-POINT asks the operator to press **YES** to ignore the failure and continue, or press **NO** to stop.

c Modify CHEK-POINT to work on problem areas only, such as large SMT programmable gate arrays.

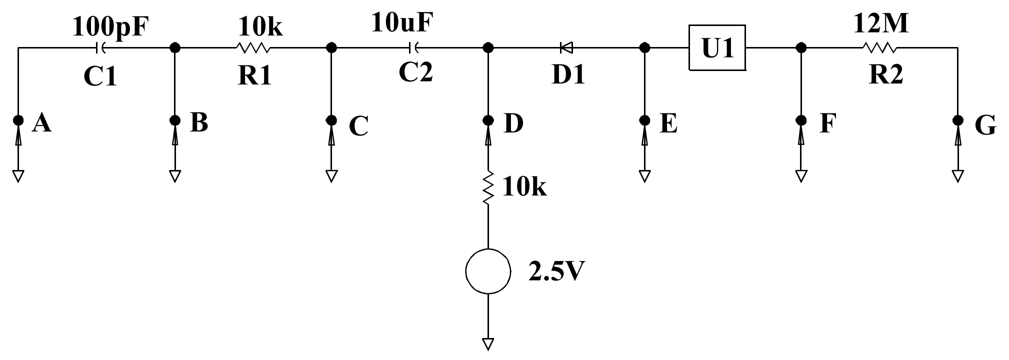
To do this, modify the `pins` file or write a new one to include the nodes corresponding to the problem area of the board or the device.

As shown in **Figure 7-1** on page 7-11, CHEK-POINT applies a 2.5 volt source to each node through a 10K ohm resistor. As the source is applied to one node, all other nodes are connected to ground. CHEK-POINT expects a leakage path through the board under test. The resistance of the path is calculated by the current through the 10K ohm resistor. Any resistance less than or equal to 1M ohms is considered a connection.

In some cases the 2.5 volt source will reverse bias a diode. If this diode is the only possible leakage path, CHEK-POINT sees an open. To avoid invalid failures, any time CHEK-POINT sees an open it reverses the polarity of the source. This would forward bias the diode and there would be a current path.

Problems can occur with small capacitors with one lead not connected to anything or with resistors greater than 1M ohms with one end not connected to anything. See C1 and R2 in **Figure 7-1**. To correct this problem, remove these nodes from the `pins` file.

Figure 7-1 CHEK-POINT



d Add a `report limit <#>` statement as the first line in the pins file to instruct CHEK-POINT to stop listing failures after the specified number.

This prevents a very long report when the entire board is not contacting the fixture.

The failure report for CHEK-POINT shown in **Example 7-6** lists the failure, brc, node name, and all device.pins on that node.

e When you edit the `pins` file you must re-store it and run Develop Board Test.

Example 7-6 CHEK-POINT report

```
-----------------------------------------
    CHEK-POINT Report for "pins".
    Thu Dec 29 12:41:34 1988
    07980-66521 11/03/87 Rev A3 BOARD
      Tue Nov 03, 198710:08 am

-----------------------------------------
    Failed Open #1
    (20476) C68-2
            C68.2
            C69.1
            R22.2

-----------------------------------------
    Failed Open #2
    (21410) U30-3
            U30.3
    -----End, 2 Problems Reported-----
```

5 Add custom ease-of-use functions for the operator.

NOTE

You can add custom operator capabilities by programming the softkeys in the testplan, as described in Chapter 3, **The BT-BASIC Environment** of *Test Development Tools*.

The customized keys are made available to the operator when the Stop key is pressed. You can also program the boxes in the operator keypad to execute BT-BASIC commands on the command line. Directions for implementing these two features are included in an online application note available in the system software.

6 Enter a temporary programming session.

When the testplan is idle, you and the test engineers can use the program monitor softkey or box in the operator keypad to temporarily log into the system and access any of the programming software. See **Table 7-3** on page 7-13. You must use a valid user login. Operators cannot log in unless they are given a valid login id and password. To exit and return to the testplan idle state, type `test monitor` on the BT-BASIC command line. When you exit the programming session, it automatically shuts down.

When you are in the temporary programming session, you are free to perform any normal user functions, but from the one window only. You can run any program from the command line and can open other BT-BASIC windows.

Table 7-3 Program monitor mode statements

Statement	Description
basic; window	Initialize a new BT-BASIC window in **basic** mode
digital; window	Initialize a new BT-BASIC window in **digital** mode
execute; window	Initialize a new console window running the shell
get digital/u1; window	Initialize a new BT-BASIC window with the test for **u1**

7 Maintain the test fixture.

Test fixtures require maintenance and possibly repair after prolonged use. Common problems include:

- worn or bent probes
- worn gasket material
- vacuum leaks around the board under test
- bent tooling pins

NOTE

To maintain or repair your test fixture, refer to the instructions in *Building Board Test Fixtures* documentation.

Implement Changes

1 Implement necessary changes.

You might need to modify the test, fixture, or both, due to engineering changes during the life of the board. Some of the changes that can be required are:

- Changing the value of a device.
- Adding or removing a device.
- Moving an existing device to a new board location.
- Removing a probe location.
- Cutting a PC trace, possibly creating two nodes from one.

To implement these changes, edit the proper files and run the Develop Board Test function of IPG Test Consultant.

2 Mark customs and modified tests as permanent.

To prevent IPG from over-writing your custom and modified tests in the future, you should mark them as `PERMANENT` in the `testorder` file. If you need to run IPG incrementally, you can easily specify which tests to re-generate and update the `ipgtc.tests` and `testorder` files.

a Run IPG Test Consultant and Develop Board Test.

b Click on the Generate Tests Using Agilent IPG task to display the action list and select **Modify List of Tests to be Re-generated**.

c Use the displayed form and the mouse to move the tests to the appropriate fields.

d Press the **SAVE** button to update the `ipgtc.tests` and `testorder` file.

e To prevent IPG from over-writing your modified testplan, add the `testplan generation off` statement as the first line in the `testorder` file.

IPG Test Consultant and Board Consultant not only generate new tests and fixtures, they manage any changes you need to make either before or after the fixture is built. Anytime you modify a file, execute Develop Board Test to update all files and reports affected by your modification. Enter Develop Board Test from the main menu of IPG Test Consultant to force it to evaluate the necessary changes.

NOTE

See Chapter 1, **Agilent IPG Test Consultant** in *Test Development Tools* for information on IPG Test Consultant; see Chapter 2, **Agilent Board Consultant** in *Test Development Tools* for information on Board Consultant.

Index