Technical Guide
to
IntelliPort-II
&
IntelliPort-IIEX
Diagnostics

Version 3.02.0

23 September 1999

Procedure number: DRAFT (was M037 - Rev A.)

This manual is part number DRAFT (was 0-20045)

Introduction

The IntelliPort-II family of products consists of multiport serial communication cards designed for ISA, EISA, Micro Channel, and PCI buses. These products support from four to eight serial ports per card. The IntelliPort-IIEX family supports from eight to sixty-four ports per card, using daisy-chained expansion boxes. Despite this diversity, all share a common architecture designed to present a uniform host interface.

This uniformity in design allows common drivers, interface libraries, diagnostics, and trouble-shooting techniques to apply to all products in the IntelliPort-II family.

This manual explains the tools currently available for both hardware quality inspection and troubleshooting, with an in-depth explanation of the tests, and technical tips.

General Considerations

The principle diagnostic is called I2DIAG.EXE, which runs under the MSDOS (3.xx) operating system. This o.s. was chosen as a test environment for many reasons. MSDOS is inexpensive, widely available, and compatible with almost every 2/3/486-based machine. Most other hardware components designed to run in these hosts use MSDOS as a testing environment. Finally, the low disk and memory requirements for MSDOS make it suitable for the manufacturing environment.

Because these cards are frequently used to connect several terminals to a host, they are usually installed under UNIX, XENIX, or some other multi-user operating system. Does testing under MSDOS adequately predict the reliability under these other operating systems? Yes! These cards use the host's i/o addressing space and an interrupt. These resources, unlike memory addressing space, are equally accessible in real mode (MSDOS) and protected mode (UNIX).

This implies that a board which has been configured and installed under UNIX may be addressed under MSDOS in this same configuration. Any hardware capabilities which might have been verified under UNIX can be verified under DOS. Since there is less operating system overhead and more control over precise timings under MSDOS than most multi-user systems, the tests can in fact be more thorough and precise.

Except for bootstrap firmware, the programs which run on the board are downloaded from the host and run in on-board DRAM. This downloaded code, or loadware, is more easily upgraded than firmware, which requires physically removing and replacing chips on the board.

Some tests are performed using the standard loadware, which is also used in normal operation of the board. In addition, some tests download special loadware modules designed to test specific parts of the hardware.

While I2DIAG may be used to detect and diagnose most faults, in extreme cases, a board may be so severely dysfunctional that it cannot reliably address DRAM or the FIFO. In that case, it is impossible to download tests, and the cause of failure must be determined from the results of the power-on self-test, as shown by the on-board LED.

Scope & Release Notes.

Current Release: 3.02.0

This manual applies to I2DIAG.EXE version numbers 3.02.0, as well as any subsequent versions numbered 3.02.x which may be released for bug-fixes, etc. Future versions numbered 3.03.x or higher will be accompanied by manual changes.

Version 3.02.0 additions

1. Support has been added for the PCI-CEX controller, a PCI version of the IntelliPort-IIEX expandable controller.

2. Support has been added to support the IntelliPort Plus product line. These products use ST654 UARTS instead of the Cirrus 1400, and are capable of line speeds up to 921,600. This mostly involved updating the loadware (both standard and diagnostic) and adding higher line speeds to some of the test options.

   IntelliPort Plus products include the Power RackPort expansion boxes, the IntelliPort Plus ISA-8 and IntelliPort Plus ISA-4.

IntelliPort-II or IntelliPort-IIEX?

IntelliPort-IIEX normally refers to controllers which are designed to connect to one or more expansion modules, to support 8-64 ports. The controller contains the processor, memory, and host interface, and the expansion modules contain the UARTS, signal drivers, etc. In contrast, IntelliPort-II normally refers to non-expandable controllers which contain their own UARTS and signal drivers, the external box being completely passive.

When the IntelliPort-II and IntelliPort-IIEX lines were first introduced, all of the non-expandable controllers (ISA-8, ISA-4, etc.) used one type of host-interface (AMD BiFIFO), while the expandable controllers (ISA-CEX, etc.) used our own design.

Shortly after the original product introduction, the expandable bus was implemented in ASIC. Soon after that, many of the non-expandable controllers were redesigned to use this same ASIC. Therefore, *as far as the host interface is concerned*, these newer non-expandable controllers behave identically to the IntelliPort-IIEX controllers, except that they always run in 8-bit data mode.

In IntelliPort-IIEX products, the UART's, drivers, and receivers are in the expansion boxes. These boxes are presently available in 8-port or 16-port sizes. Up to four expansion boxes may be daisy-chained together (side by each), and the first box connected via short cable to the controller card in the host machine.

The signals running from the controller card are now bus signals; data, address, and control, as well as the power required by the expansion boxes. In case the +12/-12 loading is too great for your host's power supply, there is a power adapter module which connects between the cable and the first box.

This means that an IntelliPort-IIEX controller, with expansion boxes attached, can support from 8 to 64 ports. Either an 8-port or 16-port box can be used anywhere in the daisy chain. The convention for numbering the ports is that first box contains ports 0-15 or 0-7. The second box contains ports 16-31 or 16-23, regardless of whether the first box was an 8-port or 16-port. And so it continues. This means that when 8-port boxes are used, there may be ports "in the middle" that are missing. For example, if you connect a single 16-port box, you will have 16 ports, numbered 0-15. But if you connect two 8-port boxes, you will have again 16 ports, but they will be numbered 0-7 and 16-23. The advantage of this is, if you have several boxes and change some from 8-port to 16-port, you don't have to re-number the remaining ports.

These facts are important to remember when using the C command-line argument, to specify the channels to test.

This architecture also implies that, in some cases, no boxes might be attached. If this happens, the board has no ports whatever. But it will still perform most low- level tests, such as DRAM tests or host interface FIFO tests. This is useful to know when you are in troubleshooting mode.

While it is possible to test an IntelliPort-IIEX without any boxes attached, the test is by no means a full one. In particular, you won't know if the external bus interface to the boxes is any good. To fully exercise and test this interface, we recommend that the board be attached to at least two external boxes. This ensures that the interrupt lines (from the boxes to board) will be fully tested. These boxes could be either 8-port or 16-port. Using 16-port boxes instead of 8-port, and using four boxes instead of the two may possibly test the controller card more thoroughly (simply because the interface might be busier, and the bus longer).

The IntelliPort-IIEX features a 16-bit wide host FIFO interface, which is built from four separate FIFO chips, and external logic to support the mailbox, interrupt, and status registers. It also supports optional DMA transfers between the FIFO and the board's local DRAM. Because of these differences, there are new tests to help isolate, in case of failure, which of the FIFO's chips is bad. Also, there is a new tests designed specifically to test the DMA operation. *These tests also apply to the "ASIC" versions of our non-expandable controllers.*

The original IntelliPort-II used a single 8-bit AMD BiFifo chip which contains the host interface FIFO's in both directions, as well as all the mailbox, interrupt, and status registers.

Of the IntelliPort-IIEX controllers, the ISA-CEX alone can be configured optionally to use an 8-bit host interface. You would not normally do this, because the bus transfers on an ISA machine are much faster in 16-bit mode than in 8-bit mode, besides requiring roughly half as many of them.

We included the 8-bit option in case you were installing the card in a hypothetical (and as yet undiscovered) host which did not properly implement the 16-bit i/o timings. In such a case, the 8-bit timings would be looser, and more likely to work. This also lets you put the card in an 8-bit slot, so long as you avoid using interrupts 10-15, and the edge connector doesn't collide with any components on the motherboard.

To configure the ISA-CEX for 8-bit operation, turn DIP switch #8 OFF, or install the card in an 8-bit slot. For normal testing, the card should be in 16-bit mode. Install it in a 16-bit slot, and ensure switch #8 is ON.

There is little need to test ISA-CEX boards in 8-bit mode. They will be shipped in 16-bit mode, and very seldom should be changed. Eight-bit operation is mostly a subset of 16-bit operation, except for the logic that enables the board to select between the modes and report the selection to the host. It might be worth checking this on a board from each lot, just to make sure no systematic fault has crept in.

Bus Differences

IntelliPort-II and IntelliPort-IIEX controllers are provided for three host bus architectures, ISA, EISA, and Micro Channel (MCA).

ISA controllers:

Address selected by DIP switch.

Edge-triggered Interrupt selected under software control.

May be installed in EISA or ISA machines. EISA configuration files are provided for these boards when installing in an EISA host. This allows the EISA configuration utility to determine available interrupts and I/O addressing.

EISA controllers:

Address determined by slot selection.

Level-triggered Interrupt selected using the EISA configuration utility and Computone-supplied configuration files.

May be installed in EISA machines only.

MCA controllers:

Address selected using the Micro Channel configuration utility and Computone-supplied configuration files on the “Reference Disk”.

Level-triggered Interrupt also selected using Micro Channel configuration.

May be installed in Micro Channel machines only.

PCI Controllers:

Address and Interrupt are selected automatically by the host’s PCI bios. Any drivers or test software must use these pre-selected values and should not attempt to override them. The particular values chosen will depend on the host machine and on whatever other cards are installed in it.

Level-triggered Interrupt

Must be installed in PCI slot.

Bootstrap Proms and Power-on Self-Test

Before learning the host-based diagnostics, it is important to understand what is done by the bootstrap proms, and how to use the on-board LED to determine the cause of failures.

The current status of the board is always indicated by the LED found at the rear of the board, near the connector(s). This LED contains both a red and a green light source, and so can be in one of four states: off, red, green, or yellow (green and red mixed).

Normally, it flashes yellow while the board is waiting for loadware, and flashes green while standard loadware is running. A flashing sequence which includes red indicates the board has detected some unrecoverable error. Each sequence is a particular code, and indicates a specific condition.

LED error codes are numbered, and listed in appendix A. The number is indicated in this way: First there is a single red flash. Then the error number is given through a series of green and yellow flashes. The green flashes count ten, and the yellow flashes one. Then the process repeats. For example, error code twelve would be indicated red, green, yellow, yellow.

Error codes one through five are reserved for use by the bootstrap firmware. Higher codes are used by the standard loadware. See appendix A for a complete list.

Once the board is in a fatal error condition, it will no longer communicate with the host through the FIFO. Only a board reset (power-up, bus reset, or under host-software control) can bring the board out of this condition.

If any unexplained fatal error is displayed, before you reset your system, record the code and try to determine what was occurring just prior to failure. Notify your Engineering contact for assistance.

If there are serious defects in the board, the bootstrap code may detect a fatal condition, or fail to run at all. When this occurs, knowing the sequence of events at power-up time is helpful for diagnosing the problem.

Sequence of Events at Board Reset

Although the IntelliPort-II and IntelliPort-IIEX boards use different bootstrap firmware, the sequence of operations is almost the same, except as noted below.

1. The LED is cleared (off) by the power-up or reset pulse. When the on-board processor receives a reset, it will begin to fetch code from the bootstrap proms. If the processor is quite dead, or unable to fetch even the bootstrap code, the LED will remain off. The only exception would be an unlikely failure in the LED's reset circuit itself. This condition could make the LED power up (and remain) in any state.

   (Technician's Hint: If the LED remains unlit, it almost always means that the processor is not running. Is the processor clock running? Is the processor inserted correctly? re the bootstrap proms good? Is the LED itself defective?

2. The bootstrap code initializes the processor's internal registers, allowing it to write the LED control register, and enabling the DRAM refresh. The LED is immediately turned yellow.

   (Technician's Hint: The enabling of DRAM refresh is another benchmark event. Once this has been enabled, the processor will periodically read DRAM locations, once every few microseconds. This refresh should take place even if the processor later fetches bad code and halts. Therefore, a complete lack of any DRAM accesses once the LED is yellow suggests problems with the CPU internal refresh generation or the DRAM state machine.)

3. The host interface FIFO is reset and a basic FIFO diagnostic performed: Do the status bits indicate when the FIFO is full and when it is not? This is intentionally an easy test. If the status bits do not track, either the FIFO is very sick or the address decode circuit is not even accessing it properly. There is no hope of downloading any useful tests from the host, let alone the standard loadware. We issue a fatal error code 1 to the LED.

4. Next, other hardware is checked:

   On IntelliPort-II boards, an on-board resource register is read which tells how many UART's are supposed to be present. The firmware confirms this by reading a valid revision number from each chip.

   On IntelliPort-IIEX boards, the chip-mapper and UART's are within each expansion box, not on the controller. The configuration register is read to determine how many boxes (if any) are attached to the controller. Then a resource register on each box is read to determine the number of UART's it supports, and whether there is an LCD controller present. As with the IntelliPort-II, the firmware confirms the expected UART's are present.

   If any discrepancies are found, the situation is not fatal, but is reported in the Power-On Reset Message which the board will issue to the host.

5. Then DRAM is tested, using a pseudo-random pattern test. If this test fails, we issue fatal error code 2 on the LED.

6. After passing the memory test, the on-board processor checks the status of the self-test flag. This flag is set if the bus signals /IORD and /IOWR are both low. This cannot occur in a host machine, so the processor concludes it must be in a burn-in rack or test bed.

7. Depending on whether the self-test flag is set or not:

   a. On seeing the self-test flag set, the board first turns the LED green for 1/2 second, then jumps back to step (2). The power-up self test is performed forever, the LED flashing yellow, green, yellow, green, until the boards are powered down, or until a fatal error occurs.

b. If the self-test flag is clear, then the board concludes it might be in a host machine, and it performs further tests related to the downloading of code.

8. The board determines whether a debug port has been attached. Normally, one has not. When attached, the board may use this port in lieu of the FIFO to download tests and perform other functions. The port is also used to aid in development of loadware.

   Downloading code through the debug port is not currently supported in the IntelliPort-IIEX firmware.

   At the present time, the debug port is beyond the scope of this manual, except to note that if it is attached (say, for testing purposes), the board may still download code through the FIFO.

9. The board then writes a 16-byte block of power-up status information to the FIFO, to be read by the host. The format of this block, called the “power-up message” is given in appendix B.

   This message validates to the host that there is an IntelliPort-II or -IIEX board at this i/o address, tells the type of board installed, and provides other hardware information.

   The diagnostic I2DIAG.EXE expects power-on self-test to have been completed and this message written within two seconds of reset.

   An IntelliPort-IIEX board will now send the message “Self-test Passed” to any expansion box which has an LCD controller.

10. The board now waits for the host to download some code through the FIFO. While waiting for code to be downloaded and during download, it flashes the LED yellow, off, yellow, off.

11. When all the code is downloaded, the board calculates a CRC for downloaded bytes, and compares it with one supplied by the host. If there is a mismatch, fatal error code 3 is sent to the LED.

12. If the download is successful, the downloaded code is executed, and the bootstrap process is complete.

IntelliPort-II and IntelliPort-IIEX Loopback Connectors

The requirements for most external loopback tests can be filled with a single loopback connector which connects the following signals:

Tx to Rx

DTR to DCD and DSR

RTS to CTS and RI

On the IntelliPort-II, DSR and DCD are both supplied by pin 20. On the IntelliPort-IIEX, DCD is on 20 and DSR is on 11.

On the IntelliPort-II, the RI signal is not supported. On the IntelliPort-IIEX, the RI signal is on pin 22.

A loopback connector designed for IntelliPort-II and IntelliPort-IIEX testing is available from Engineering. This loopback connector has built-in LED's to help monitor the signals.

Some other tests require loopbacks with the data-set signals "backwards" from this standard plug. That is, DTR connected to CTS and/or RI, and RTS connected to DCD and/or DSR. Generally, these other tests do not test any new functions, but may be useful for a technician trying to track down a problem. So in a manufacturing environment, you can stick to tests using the "standard" loopback pin-out.

If you select tests requiring mutually exclusive loopback-plug pin-outs, the i2diag will stop between tests as necessary to allow you to change plugs.

Introduction to I2DIAG.EXE

The complete I2DIAG diagnostic consists of the program I2DIAG.EXE and several *.LOD files which contain various loadware modules. These modules are listed in Appendix C.

The largest module, FF.LOD, is the standard loadware. This is the code which is used by the board in normal operation. Using this module, the diagnostic performs data loopback tests on multiple channels at selected or random baud rates and protocols. It also performs data-set signal loopbacks, and checks break generation and detection. These are known as high-level tests.

The remaining modules are used together with other tests to validate specific areas of hardware. These are known as low-level tests.

Low-level tests are important because the high-level tests assume a great deal of low-level functionality, i.e., good DRAM, good FIFO, good interrupts, etc. If any of these resources are unreliable, a high-level test will probably fail, but it will interpret the failure in a misleading way. For example, a missing interrupt could be interpreted as late or missing characters in a loop-back test.

(Technician's hint: Another type of high-level test is included: the ability to generate a continuous signal on either TxD, DTR, or RTS for a selected channel or channels. This is useful in tracing signals.)

To run the diagnostic from the MSDOS prompt, type I2DIAG, followed by various command-line arguments. There are three types of arguments. One is used to select which tests are to be performed and how many times. Others, called "options", select the amount of output to be generated, the baud rates and protocols to be tested, the interrupts to be used, and specific channels to test. Some options are required for particular tests, but others are... optional.

Since the operation of the test is controlled completely by command line arguments, you can create MSDOS batch files (*.BAT) to invoke I2DIAG in the optimal way for particular situations. See Appendix F for examples.

Built-in Help

When I2DIAG runs, it first checks the command-line arguments for completeness and correct syntax. For example, have any tests been selected? If a selected test uses interrupts, has an interrupt been selected as well? If any command line error is found, the test prints an error message, as well as a message telling you how to print help screens.

To view or print the help screens, type either

```
I2DIAG HELP | MORE
```

or

```
I2DIAG HELP > PRN
```

You should either print the output or pipe it to MORE, since the help message is extremely long. It summarizes the available arguments, explains the tests, and so on.

Termination

You may strike CTRL-BREAK at any time to terminate the test program while it is in progress. Allow a few seconds for the test to respond.

If the i2diag runs to completion and passes every test, it returns an exit code of 0. If it fails any test, reports command-line errors, or is aborted by the operator, it returns a non-zero exit code. This allows you to write *.BAT files to run several tests using the ERRORLEVEL n function to terminate the batch processing when errors are found.

Command-Line Arguments

Unlike many MSDOS commands, the command line arguments to I2DIAG are not preceded by a slash (/). Instead, each argument begins with a character which identifies what type of option it is. After this character, some options require additional information, such as a key-word, a number, or a range of numbers. This comes immediately after the identifier, without any spaces between. Spaces are used to separate the arguments from each other.

Some arguments require that a number or a range of numbers follow the initial character. Numbers may be given in decimal, octal, or hexadecimal (hex) notation. This is because certain numbers are normally expressed in decimal, and other information (such as the board address) is normally expressed in hexadecimal. To express a number in hex notation, prefix it with 0x. For example, the default address of an ISA-8 would be written as 0x308. To express a number in octal, prefix it with 0. Otherwise, it is taken as decimal. Two numbers separated by a hyphen indicate a range.

Some arguments require specific keywords to follow the initial character. An example is the hyphen (-) argument, used to select communication protocols for the high-level tests. To specify character size, for example, you must type either -7 or -8. Allowable keywords are listed below with their arguments, when appropriate.

There are three types of arguments; tests, options, and protocols. Tests are indicated by T, protocols by a hyphen, and other options by anything else.

T to select Tests

The T argument specifies which test to perform, and optionally, the number of passes. The test is identified by a number (see the next section). A single test can be selected by using a single number, or several tests using a range of numbers. If a single test is specified, the number of passes for this test can be given by putting a decimal point immediately following the test number, followed by the number of passes. If a range is used, or the number of passes is not given, a default number of passes is assumed. This default differs from test to test.

For example T3.10 selects 10 passes of test number 3. T1-9 selects the default number of passes for tests 1 through 9.

There may be many T arguments on the command line. Tests are performed in the order listed. A range is treated as though the tests were listed separately, in order. If the same test is selected more than once on the command line, it is performed only once, in the order it first appears. The number of passes, however, is taken from the last time it appears. This allows you to specify a range of tests, and then override the default passes by using additional arguments. Note, that if you specify zero passes, the test is skipped.

For example T10 T3-11 T4.100 T5.0 performs, in order, tests 10, 3, 4, 6, 7, 8, 9, and 11, using the default number of passes for all except test 4, which will do 100 passes. Test 5 is skipped entirely.

You cannot specify a range and override the default number of passes in the same argument. For example, T10-13.5 is not allowed because the intent is unclear. Exactly which tests are to run for five passes? All of them or just the last?

Each test has a maximum number of passes associated with it Generally, this maximum will be either 1 or a very large number. Tests having a maximum of one pass are generally special purpose tests (see Appendix E). If you specify a number of passes greater than the maximum for a particular test, i2diag will pretend you specified the maximum, and proceed without reporting any errors.

The tests numbers presently range from 1 to 51, but some numbers in between are skipped. These numbers are reserved for future tests. If we add a new test, we would like to group it with other similar tests, so that you can specify ranges of tests whenever possible. By leaving “holes” for them, we won’t have to re-number lots of tests later, and you won’t have to change your *.BAT files. (As was necessary with the 2.01 release).

If you include a reserved test number in a range of tests, it will just be skipped. But if you put it by itself on a command line, the program assumes you actually wanted that particular test. It will report that you have chosen a reserved test number, and stop.

For some tests, the number of passes actually represents seconds. This is true of most tests which were designed as exercisers. For example, tests which continuously toggle data set signals, or transmit data are really designed to exercise the port, not to test it. The technician, not the program, determines whether all is going correctly. So for such tests, it is more uniform to specify the absolute time it shall take, instead of the number of passes.

All the tests are described in appendix E.

- (Hyphen) to select protocols

Most of the high-level tests involve transmitting and receiving characters. Usually, the transmitted data is somehow looped back to the receiver, read, and compared with what was transmitted.

This serial data is characterized by five parameters collectively called the protocol: character size, number of stop bits, parity, baud rate, and flow control. Normally, when each pass of a high level test is performed, the test program picks these five protocols at random for each channel being tested.

For example, when running an external data loop-back test, one channel may be running at 300 baud, 7 bits, even parity, 1 stop bit, xon/xoff. At the same time, another channel may be running at 76800 baud, 8 bits, odd parity, and two stop bits. This random selection is done to help detect any unexpected interactions between the channels.

Using the hyphen arguments, you may override this random selection for any or all of these five parameters. Any unspecified parameter will still be chosen at random.

For example, the protocol selection -9600 -odd -1 -x specifies that all passes of all tests on all channels will be performed at 9600 baud, odd parity, one stop bit, xon/xoff. The character size will still be randomly chosen, either seven or eight bits.

A list of all the protocol arguments is given in Appendix D. These are also the choices from which the random selections will be made when the protocol argument is not given.

One use of the protocol options is with the transmission test. Often, the output will be connected to a terminal for validation. You will want the protocol to match that of the terminal. Even when using the transmission test without a terminal connected, the signal will be easier to trace if the baud rate is known ahead of time.

Another use is with the data loopback tests. This can help reproduce suspected problems which seem to be linked to high (or low!) baud rates. Or, it can force larger volumes of data to be tested in a shorter time by limiting the testing to a single high rate.

When running a data loopback test without specifying any protocols, the protocols are chosen at random for each channel every few passes. If a slower baud rate is chosen, fewer characters are transmitted on each pass. In this way the channels using a higher baud rate will not waste so much time waiting for the slower channels to complete their passes.

I to select Interrupts

Most of the low-level tests do not require the board to generate interrupts. The exception is the interrupt line test, which can test any or all of the interrupt lines supported by the board. The high-level tests, on the other hand, require a single interrupt, and assume it is good.

The I argument takes a number or range of numbers to specify which interrupt or interrupts will be tested. There is no default value assumed.

This argument may appear more than once. The first interrupt specified will always be used during the high-level tests (if any are selected). All interrupts specified will be tested by the low-level interrupt test (if it is selected).

Micro Channel and EISA boards

You should use the BM or BE argument when testing Micro Channel or EISA boards.

If the BM or BE arguments are used, the I argument is not required. The test will use whatever interrupt you configured earlier using the Micro Channel or EISA configuration utility. If you do use the I argument to specify one or more interrupts, the test will verify that this matches what had been configured. If the interrupt test (T11) is chosen, the test will temporarily change this configuration in order to test all the interrupt lines you have specified.

PCI Boards

You should use the BP argument when testing PCI boards.

If the BP argument is used, the I argument should not. The tests always use the interrupt that was selected at bootup time by the system's PCI BIOS. (On a PCI host, interrupt selection is the responsibility of hardware on the host, not on the adapter card. The adapter card deals with only a single interrupt line, so from its perspective, one interrupt is the same as another. On the other architectures the adapter card has a separate line for each interrupt. In principle the card could have a defect that affects one line but not another. That is why testing multiple interrupts with T11 is not required for PCI, but is desirable for the other bus types.

ISA Boards

You must use the I argument for any of the high-level tests, and for T11. Since no pre-configuration is required, any available interrupt(s) can be tested. (Note however, if you are testing an ISA card in a PCI machine, you may need to run the BIOS configuration utility to reserve one or more interrupts for ISA use.) It is your own responsibility to avoid choosing interrupts where there may be a conflict with other hardware.

General Considerations

The supported interrupts are 3, 4, 5, 7, 10, 11, 12, and 15. If a range is given, it is taken to include only the supported interrupts. For example, the arguments I10 I3-15 will select IRQ 10 to be used for all the high-level tests. For non-PCI controllers, the interrupt test T11 will test all supported IRQ's (excluding, for example, IRQ 6).

When a range is given, each limit of the range should be a supported IRQ.

A to select the board Address

The A option is used to specify the base i/o address of the board to test. This defaults to address 0x308, so normally this option is not used.

The A option allows testing of a board in systems having other hardware at the default address.

ISA Boards

Since ISA boards do not auto-configure, you must remember to set the dip- switches accordingly.

For ISA boards, DIP switch positions 1-7 correspond to address lines SA9-3, respectively. An open switch selects binary one, a closed switch zero. For example:

Argument	**DIP setting (1 = open, off)**
A0x308	1100001(default: no A argument needed)
A0x310	1100010
A0x318	1100011
A0x320	1100100

Micro Channel Boards

For Micro Channel boards, the address is configured using the reference diskette for that machine, and an option file which we supply on the Computone Option and Configuration Diskette. It also defaults to 0x308.

Micro Channel supports a wider range of i/o addresses than the ISA bus, and our MC controllers support many of these addresses. If the command line option BM (Bus Micro Channel) is given, the test will allow these higher addresses to be specified. Otherwise, anything above 0x3f8 is rejected out of hand.

If the command line option BM is used, the A argument is not required: the test will use whatever address you have configured. If the A argument *is* used, the test will confirm that it matches the configured address.

If neither the A nor the BM arguments are used, the default address, 0x308, is assumed.

EISA Boards

The EISA-8 and EISA-CEX use only i/o addresses from the EISA slot-dependent range. A board in slot 1 will have a base address of 0x1c88, a board in slot 7 will be at address 0x7c88, and so on. Use the command line option BE to allow these higher addresses to be given.

If the command line option BE is used, the A argument is not required: the test will use whatever address you have configured. If the A argument *is* used, the test will confirm that it matches the configured address.

If neither the A nor the BE arguments are used, the default address, 0x308, is assumed.

PCI Boards

The address of a PCI board is assigned automatically at boot-up, by the PCI BIOS. This makes it rather difficult to anticipate what this value might be in advance.

If the command line option BP is used, the A argument is not required: the test will use whatever address was assigned by the PCI BIOS. If there A argument were used, the test would confirm that it matched the one selected by the PCI BIOS. Unlike the Micro Channel or EISA cases, you cannot configure or force the board to be mapped into any particular address. You are forced to use whatever the PCI BIOS has chosen, and by definition if there is a discrepancy, *you* are the one who is wrong, not the BIOS. So there is little point in using the A argument for PCI boards.

If the BP option were omitted, then the test will try to use the address given by the A argument (or 0x308, if the A argument were not present. Guessing the correct address to supply with the A argument is impractical except in special circumstances.

General Considerations

For any type of board, the address must be a multiple of eight, or it is rejected.

While I2DIAG only tests a single board at a time, this option allows you to load multiple boards into your test system at once, at different i/o addresses. Power up the system. Use the A option to select the board to test. While this saves power-up time and machine cycling, care must be taken not to confuse the boards being tested, and to restore the DIP switches to appropriate settings.

Be cautious when using this argument. Do not attempt to run the test if there may be some other type of hardware at the selected address. Even the act of trying to read the FIFO power-up message will have unpredictable effects on hardware other than IntelliPort-II boards.

C to select Channels

The C option is used to specify which channels to test. A number or range of numbers may be given. If this option is not used, all channels supported by the product or configuration will be tested. For example, when testing an ISA-8, eight channels will be selected for test. If one of the UART's is defective, the test will not mistake this situation for a good ISA-4 board.

If the C option is used, the test verifies that all selected channels are supported by the hardware under test. If unsupported channels are specified in the option, this is reported as an error.

(Technician's hint: this is useful with the tests designed for tracing signals, to limit the exercise to the single channel in question. It is also useful when there has been a failure, to limit your attentions to the offending ports.)

Another possible use would be manufacturing test. The batch files for testing a particular product could hard-code the expected number of channels for a particular product. The batch file used for testing ISA-8's could contain C0-7 while the file for ISA-4's would contain C0-3. The batch files for the IntelliPort-IIEX family would differ according to the configuration of boxes being tested. For example, an ISA-CEX with three 8-port boxes attached would use C0-7 C16-23 C32-39.

If the C argument is not used, the test determines which channels to test using information in the FIFO power-up message. This in turn comes from the bootstrap firmware reading a hardware configuration register. If this register were unreliable, you could test an ISA-8 according to the more modest requirements of an ISA-4. In the IntelliPort-IIEX, a hardware failure could possibly cause a connected box to be undetected.

Since the number of channels to be tested is normally printed, a technician would spot such discrepancies. When dealing with large numbers of boards in a manufacturing test environment, this sort of problem is easier to miss, and specifying the channels provides an extra check.

When using batch files having the C argument, be sure to avoid the opposite error, viz,: if you run the *.bat file for testing ISA-4 boards (containing C0-3) when actually testing an ISA-8, only the first four channels will be tested, and no error is reported.

B to specify the Bus architecture.

In normal operation, IntelliPort-II and IntelliPort-IIEX controllers for ISA, EISA, and Micro Channel buses work in much the same way. They differ only how the i/o address and interrupt are chosen and configured. One these are known, the drivers function exactly the same regardless of the type of bus. This is in fact an important design goal, as it reduces driver proliferation.

For certain diagnostic tests, a knowledge of the bus architecture is important. For example, if we are testing a Micro Channel card, we would like to check its POS registers; these registers do not exist on ISA or EISA cards.

If this argument is not used, i2diag assumes it is running on an ISA machine.

BI	You are running on an ISA machine. This is the same as saying nothing at all. Board addresses are limited to 0x3F8, and interrupts are selected by commands sent to the loadware.
BM	You are running on a Micro Channel host. Before any testing starts, the test reads the adapter I.D. registers for each slot, and identifies any MC-8 or MC-EX controllers present. If an address was specified using the A argument, it determines whether a board a board is present configured for that address. If an interrupt was specified using the I argument, it verifies that the selected board was configured for the selected interrupt. If the A argument was omitted, the test will select the first board found through the POS registers. If the I argument is omitted and high-level tests are specified, the interrupt configured into the POS registers will be used. If the low-level interrupt test was specified, the I argument is required. If multiple interrupts are specified, the test ensures that the first interrupt listed matches the one configured in the POS registers. During the test, the POS registers are re-configured to allow the board to test all the selected interrupts, after which the original configuration is restored.
BE	You are running an EISA controller on an EISA host. Before any testing starts, the test reads all the slot-specific ID registers and identifies any EISA-8 or EISA-CEX controllers present. If an address was specified using the A argument, it determines whether a board a board is present configured for that address. If an interrupt was specified using the I argument, it verifies that the selected board was configured for the selected interrupt. If the A argument was omitted, the test will select the first board found through the ID registers. If the I argument is omitted and high-level tests are specified, the interrupt configured into the POS registers will be used. If the low-level interrupt test was specified, the I argument is required. If multiple interrupts are specified, the test ensures that the first interrupt listed matches the one configured in the slot-specific configuration register. During the test, the configuration register is modified to allow the board to test all the selected interrupts, after which the original interrupt selection is restored.
BP	You are using a PCI controller on a PCI host. The interrupt and address are assigned automatically by the host's PCI BIOS. If the I or A arguments are given, the test will verify that they match what the PCI bios has assigned. Since it is impractical to know this assignment in advance, you should omit the A and I arguments, and the test will use the assigned values (which he learns via the PCI BIOS)…

E to specify Expandable/Non expandable.

The question is, are we testing an IntelliPort-II card or an IntelliPort-IIEX? Are we using an 8-bit or 16-bit data path?

Normally, i2diag can determine what type of card is present once it has read the board's Power-On Reset Message. And if you do not use this option, that is what it will do.

There are some tests, however, which are so low-level they do not even try to read this message. Such a test will have to know ahead of time what type of board it are dealing with.

With the E options you can specify whether you expect an IntelliPort-II, an IntelliPort-IIEX using an 8-bit host interface, an IntelliPort-IIEX with a 16-bit interface, or an IntelliPort-IIEX with either size interface.

EN	The board must be an IntelliPort-II controller. (*pre-ASIC version only.: see page 4).*
EX	The board must be an IntelliPort-IIEX controller, but may be configured for either an 8-bit or 16-bit host interface.
EX8	The board must be an IntelliPort-IIEX, configured for an 8-bit host interface.
EX6	The board must be an IntelliPort-IIEX, configured for a 16-bit host interface.

If you use an E option and have selected tests which do read the Power-On Reset Message, then it will confirm that the board you are testing meets your expectations. If it does not, an error message is displayed, and the test stops.

This makes the argument useful for manufacturing checkout. For example, if you use the EX6 argument, the test will thwart any attempt to test an IntelliPort-II board or an IntelliPort-IIEX configured for 8-bit access.

O to control Output

If this option is not used, the “normal” amount of output is shown: Tests are listed as they are begun, and if there are errors they are reported in summary form after each test.

To use this option, the O should be immediately followed by one or more letters. Each letter turns on a particular type of output. If the O appears alone, almost all output is turned off. Only errors are reported, or successful completion.

Using the O option alone to minimize output is useful when performing lots of tests over a long time. This enables as much general information as possible to remain on the screen. Using it to get additional output is useful when there are errors, and you want to get as much information as possible to determine their cause. The possible arguments to the O option are as follows:

N	Displays Normal messages. If the O option is not used at all, it the same as using ON. Normal messages include displaying the name each test before it is run, and “passed” as it passes. For certain tests, the FIFO size or amount of DRAM present on the board is listed. Without N, only the names of the tests that fail will be displayed, and a brief description of the failure.
E	Displays detailed Error reports. If the low-level interrupt test fails, the number of interrupts issued vs. caught is shown. If a data loopback test fails, the protocol used and the data written vs. read is displayed, as the channel fails. If the break loopback fails, the number of breaks sent and received are displayed. If any data-set signal loopback fails, the state of the output and input signals is listed.
P	Displays Protocols used in the high-level tests. Whenever a channel is set to a particular protocol (baud rate, number of bits, etc.) that protocol is displayed, even if there are no errors. If both the E and P output controls are used and there are errors, the protocol is listed only once.
S	At Start-up time, it will display a list of all the tests you have selected, plus all the interrupts you have selected to test (if any), plus a map of all the channels you have selected to test (if any). If you are testing a Micro Channel board and used the BM argument, it will display the slot number, adapter id, interrupt, and i/o address which were stored in the board’s POS registers.
I	Displays a map of Interrupts issued vs. caught when the low-level interrupt test is run, even if the test passes. In that case, for each interrupt the number caught will equal the number issued.
C	Displays a map of the Channels found on the board, if the C argument was not specified. Do not confuse the C argument, which specifies the channels to be tested, with the C output control defined here.
V	Displays the name and Version number of any loadware being written to the board. Also displays the 16-byte Power-On Reset Message the first time (only) it is read from the board.
T	Displays the total testing Time. For the Data Transmission test, it also displays a count of total characters sent to all channels. On completion, it waits until all characters have actually been transmitted from the board, and then reports the total number of characters sent and the time of that test.
F	Displays additional FIFO test data, in case of any failures.

Miscellaneous Options

P to repeat the entire test (Passes)

The P argument takes a number, and allows you to repeat the entire list of tests that number of times. All of the selected tests are run using the order and number of passes specified with the T arguments. When all tests are complete, the entire group is repeated as many times as needed.

Each repetition is called a main pass. When the P option is used, the number of the upcoming main pass is displayed on a line before the first test.

For example, the arguments T1.3 T10.2 P2 will run the following tests: test 1 (three passes), test 10 (two test 1 (three passes), and test 10 (two passes), in that order.

If any error occurs on any test, testing will end when the current main pass is done. This is to prevent errors from scrolling off the screen.

Y to suppress prompts (Yes!)

The Y option is used to suppress unnecessary operator prompts. For example, certain tests require loopback plugs be attached or other provisions made. Normally, I2DIAG will inform the operator separately of each special requirement and await a response before continuing. The Y option eliminates these prompts.

Some selected tests may require different loopback connectors. Should connectors need to be changed between two tests, an operator prompt will occur in spite of this option.

, to pause for errors (comma)

If the comma (,) argument appears on the command line, the test will pause and prompt for operator input after any error is reported.

L to download Loadware only.

If the L option is given, but no *comma* argument, the test will download whatever loadware is required for the selected test, and then exit. If the L and comma arguments are both present, then the test will download the loadware and then pause for operator input, after which the test will continue.

F to stop the test at the first error. (Fatal)

The F (fatal) option is used to terminate the test as soon as any error occurs. For low-level tests, this happens normally. But for high-level tests, the failure of one channel on a particular test does not normally prevent other channels or other tests from being run.

X to eXtend testing after first error.

The X argument modifies the meaning of the F argument. If a fatal error occurs, the individual test will stop, but the full cycle of selected tests and passes will continue.

M or MM to Mark errors.

When running data loopback tests without either of these options, if there is a UART overrun, parity, or framing error, the error will not be reported as such Since the test compares all the received characters to what was sent, an error would still be detected, but you would know only that a character is missing, or different.

If you use the M argument, any character marked with a parity, framing, or overrun error from the UART, will be displayed as a _ character.

If you use the MM argument, such errors will be reported in special status packets. If the loopback fails, the test reports whether there were overruns, parity errors, and/or framing errors reported.

G to specify a loG file

The G argument takes a file name. Any error reports will be written to the file as well as to the screen. Normal output, indicating passes, etc., will not be logged. This option is useful for logging the results of extended tests.

Z to increase delays (Zleep)

This is useful in determining whether a failing board is simply slow, or is failing to perform the desired function altogether. The argument may appear multiple times (separated by spaces). Each time it appears, the length of all delays is doubled again.

Note that this argument should not be necessary to adjust for host speed, since all timings are derived from the host's real-time clock, not from "spin loops" and other CPU-dependent methods.

W to Wait for Expandable Boxes

The W option is designed for use with in-house fixtures that allow expansion boxes to be safely connected and disconnected from the controller cards, without powering down the host computer. When given, pauses for operator input between each of the major passes specified by the P argument. Also, if power on test message reports no ports (i.e., no boxes are attached), the test will automatically retry until some are present.

The number of boxes expected will be equal to the number of times the W argument is given. (Remember that the W's have to be separated by spaces).

+ To specify test data

This option allows you to specify the test data to be used for certain tests which would ordinarily generate random test data.

For T15 (FIFO Block-Message Loopback Test), use **+***nnnn*, where *nnnn* is a numerical argument expressed in decimal, hex, or octal. For the data loopback tests, use **+***string.* The string you enter will be used as the loopback test data.

Appendix A: Fatal Error Blink Codes

Format: Red Light flashes once: Green light = 10,Yellow light = 1

Example: Error code 12 = Red, Green, Yellow, Yellow

Table 1: LED Blink Codes (Green=10, Yellow=1)

Number	Description
Codes issued by the bootstrap firmware:	
1	**Bad FIFO**: Full/empty flags do not properly track state of FIFO; internal registers bad.
2	**Bad DRAM**: DRAM has failed a simple random-pattern test.
3	**Bad checksum during download:** Unreliable FIFO? Corrupted loadware file?
4	**Bad product-ID register** (IntelliPort-IIEX only)
5	**Dead UART** (IntelliPort-IIEX only)
6	**Bad Mailbox** (IntelliPort-IIEX only)
7	**Bad burn-in board**: This code is issued by the special ISA-CEX burn-in firmware if they detect that the card is not connected to a functional burn-in board. This error probably implies you have mistakenly used the burn-in firmware when you should be using the bootstrap firmware.
Obsolete and reserved codes:	
6	**Dead UART:** Unused after loadware version 1.0.2 1.0.2 and earlier: Second 1400 (on ISA 8) fails to respond to reset.
8	Unused
	Codes currently issued by standard loadware
9	**Invalid Interrupt:** Board processor receives an unexpected interrupt vector. Software bug? Bad DRAM? Internal CPU problem?
10	**Bad first command from host:** The very first command sent to the board after writing the loadware must be the “set interrupt level” command. This error indicates a different command was sent first, or an invalid interrupt was requested. Driver or loadware bug? Unreliable FIFO?

Table 1: LED Blink Codes (Green=10, Yellow=1) (Continued)

Number	Description
11	**Bad Packet Count:** The host has sent a packet whose count field is 0.
12	**Invalid command number from host:** Packet contains a command number which is too large. Driver / loadware bug or mismatch? Unreliable FIFO?
13	**Bad synchronous command from host**: An unassigned command was sent, or a command was sent in a synchronous packet that is only valid in a bypass packet. Driver or loadware bug? Unreliable FIFO?
14	**Internal software check:** dss_enable() called with illegal arguments. Loadware bug? Bad DRAM?
15	**Internal software check:** Logical error during output post-processing.
16	Unused
17	**Internal software check:** List of running tasks is corrupt. Loadware bug? Bad DRAM?
18	**Bad bypass command from host:** An unassigned command was sent, or a command was sent in a bypass packet that is only valid in a synchronous packet. Driver or loadware bug? Unreliable FIFO?
19	**Internal software check:** Unable to spawn a new process (table full). Loadware bug? Bad DRAM?
20	**Internal software check:** Process stack overflow Loadware bug? Bad DRAM? Interrupt problem?
21	**Internal software check:** Transmit interrupt while break processing. A transmit interrupt was received while processing a “send break” command. This should be impossible. Loadware bug? Bad 1400 UART (Issuing interrupts when it should not be)?
22	**Incoming data or command overflow:** The host has sent more commands or data then the board has room for, in violation of the flow-control credits established between the driver and the loadware. Driver/loadware bug? Unreliable FIFO?
23	**Channel number too large:** The host has sent a packet to a channel number higher than the maximum channel for this board. Driver/loadware bug? Unreliable FIFO?
24-26	Unused
27	**Stuck Mailbox-interrupt bit:** Board has received a (mailbox) interrupt from the host, but upon servicing the interrupt, the mailbox interrupt bit is clear. Loadware bug? Unreliable FIFO?

Table 1: LED Blink Codes (Green=10, Yellow=1) (Continued)

Number	Description
28	**Dead UART:** One of the 1400 UART's does not respond to reset. Bad UART? Loadware bug?
29	**Unsupported Product**: (1.0.2 & earlier) Attempted to use this loadware on an IntelliPort-IIEX (which it did not support). **Internal Software check:** (1.0.3 & later) Unable to start up LCD process.
30	**No expansion boxes connected:** (IntelliPort-IIEX only)
31	**Reserved:** Reserved for development/debugging use: should be impossible in released product. Software bug? Bad DRAM?
32	**Invalid channel number:** The host has sent a packet for a channel which does not exist. Driver/loadware bug? Unreliable FIFO?
33	**Internal Software check:** A buffer head pointer is not word-aligned.Software bug? Bad DRAM?
34	**Internal Software check:** A buffer tail pointer is not word-aligned. Software bug? Bad DRAM?
35	**Bad 1400 chip:** Loadware timed out trying to internally loop back a single character as part of its initialization sequence.
36	**Internal Software check:** Unable to spawn the hot-key scanning task at start-up. Software bug? Bad DRAM?
37	**Internal Software check:** (Loadware version 2.0.0 and higher) Board does not have enough memory to support the number of ports it thinks it has.
38-40	Unused
41	**Invalid Interrupt:** (Loadware version 2.0.0 and higher) Interrupt received from unexpected hardware, e.g. an interrupt from the second expansion box when only one box is installed. (Earlier loadware issued code 9).

Appendix B: Format of power-up message

Table 2: Power-Up Message Format

Byte	Description		
0	0x96 (First magic number)		
1	0x35 (Second magic number)		
2	version number		
3	revision number		
4	sub-revision number		
5	product i.d — Bit-mapped:		
	Product Family, Bits 7-6 (0xc0)	0x00	IntelliPort-II (original non-ASIC version)
		0x40	IntelliPort-IIEX (including ASIC versions of non-expandable products. For example, an ASIC ISA-8 it treated as an ISA-CEX with a single 8-port box attached, limited data-set signals, and an 8-bit host interface.
	Bits 1-0 (0x03) (Valid on IntelliPort-II only)	0x01	ISA-8 or MC-8
		0x03	ISA-4
6	Bus Type — Bit-mapped:		
	Bits 2-0 (0x7) Bus type, if known.	0x0	Unknown or unspecified
		0x1	Micro Channel
		0x2	EISA
		0x3	ISA
	Bit 4 (0x10): For IntelliPort-IIEX controllers, is board configured for 8 or 16 bit operation?)	0x00	DIP Switch 8 off, 8-bit mode selected.
		0x10	DIP Switch 8 on, 16-bit mode selected.
	Bit 5 (0x20): For IntelliPort-IIEX controllers, is board installed in an 8 or 16-bit slot? (16 bit data transfers can only be done if the card is configured for 16-bit mode *and* is installed in a 16-bit slot.	0x00	Installed in an 8 bit slot.
		0x20	Installed in a 16 bit slot.
7	On-board memory size, measured in 32K byte blocks. (e.g., 4 = 128K, 16 = 512K).		

Table 2: Power-Up Message Format (Continued)

Byte	Description		
8	IntelliPort-II family:	Number of ports (4 or 8)	
	IntelliPort-IIEX family.	Bit-map of UARTS confirmed present on the first two expansion boxes: (see also byte 13). Bit 0 is set if the UART for ports 0-3 on the first box is present. Bit 1 is set for ports 4-7 on the first box Bit 2 is set for ports 8-11 on the first box… …Bit 7 is set for ports 12-15 on second box.	
9	First Byte of Power-on diagnostic results (IntelliPort-II)	0x80	Chip-mapper failed
		0x01	UART for ports 0-3 bad.
		0x02	UART for ports 4-7 bad.
10	Debug port status. (The debug port is used for development purposes and should not normally be present.)	0x00	Debug port not installed.
		0x80	Debug port present.
11	CPU Speed in tenths of MHz. For example, a value of 160 indicates 16MHz, 200 indicates 20MHz.		
12	Bits 1-0 (0x3) represent CPU type.	0x02	80c186 (All products)
		0x01	reserved for future products.
	Bit 2 (0x4) IntelliPort-IIEX only	0x00	Normal EX controller
		0x04	ISA-4 ASIC version. (Recall, ASIC non-expandables have “expandable-style” bus interfaces. The original message format did not allow for a 4-port “expansion box”.
	Bits 7-4 (0xf0) IntelliPort-IIEX only	Bit-map of expansion boxes detected: Bit 4 (0x10) is set for the first expansion box, Bit 7 (0x80) is set for the fourth box.	
13	IntelliPort-IIEX only.	Bit-map of UARTS confirmed present on the second two expansion boxes: (see also byte 8). Bit 0 is set if the UART for ports 0-3 on the third box is present. Bit 1 is set for ports 4-7 on the third box Bit 2 is set for ports 8-11 on the third box… …Bit 7 is set for ports 12-15 on fourth box.	
14	IntelliPort-IIEX: Size of Host interface FIFO.	If value of this byte is N, then the FIFO size in bytes is 2^N	
	IntelliPort-II: Not used.	FIFO is always 512 bytes.	
15	IntelliPort-IIEX only.	Number of expansion boxes installed (1–4)	

Appendix C: Download modules

Module	Test(s) used
FF.LOD	The standard loadware. Used for all the high-level tests; for example, all data loopback tests, all data-set signal tests, and all transmission tests.
FIFOB.LOD	FIFO Block-Message Loopback Test
FIFOD.LOD	FIFO Loopback Using DMA Test (for IntelliPort-IIEX only)
FIFOI.LOD	FIFO Bidirectional Loopback Test.
FIFOL.LOD	Host Interface Timing Stress FIFO Loopback Test.
FIFOM.LOD	FIFO Mailbox Test
INTT.LOD	Interrupt test
MEMR.LOD	Random-pattern Board Memory Test
MEMW.LOD	Walking-Bit Board Memory Test
NULL.LOD	FIFO Measurement Test
REGL.LOD	Interactive LCD Controller Test
REGT.LOD	Board/Box Hardware Register Test
REGT2.LOD	Debug-Port Register Test
REGT3.LOD	External Bus Exerciser

Appendix D: Protocol Options

Flow Control	-o	-x			
Character Size	-7	-8			
Stop Bits	-1	-1.5	-2		
Parity	-none	-odd	-even	-mark	-space
Line Speed	-50	-200	-2000	-9600	-64000
	-75	-300	-2400	-19200	-76800
	-110	-600	-3600	-38400	-115200
	-134.5	-1200	-4800	-56000	
	-150	-1800	-7200	-57600	
	The following line speeds are supported on *IntelliPort Plus* products only.				
	-115200	-153600	-230400	-460800	-921600

Appendix E: Detailed descriptions of tests

General Information:

High-level tests all use the standard loadware (FF.LOD), which is written to the board as soon as needed. It continues to run there throughout any successive high-level tests.

Low-level tests use special loadware modules which are loaded at the start of each test. Unlike the standard loadware, such a module is reloaded even if it is already present from the previous test

If low-level tests are listed on the command-line between high-level tests, the standard loadware will be written each time, as needed. Since this takes additional time, it is quicker to perform all the high-level tests contiguously. The test numbering is designed to encourage this, with tests T25 and higher being high-level tests, and T24 and lower being low-level tests.

If the required code module is not present in the current directory, an error is reported and that test fails.

Before downloading a module, each test performs a preamble:

1. Board is reset by writing to base address + 7.

2. Host waits two seconds for FIFO power-up message.

3. Host attempts to read Power-On Reset Message from FIFO's data port.

4. Host verifies that the Power-On Reset Message contains valid and consistent information.

(Technician's hint: The most common problem in this phase is usually something to do with a bad Power-On Reset Message. This is always fatal, and stops the test. The test will report the number of bytes read and their contents. If any are read at all, compare them with the values in Appendix B. If nothing at all is read, look at the LED. If the yellow light is flashing normally, you may have incorrectly set the DIP switch or used the incorrect address in the A option. If the light is abnormal, the power-on-self-test has failed. See earlier sections on the bootstrap process for guidance.)

When reading the Power-On Reset Message, 8-bit data transfers (from the even data-port address) are always used. The firmware will always write the message so it can be read in this way. This is because you cannot tell what sort of board you are using until the message has been read. Needless to say, all the hardware registers needed to actually read this message are kept the same among all the IntelliPort-II and -IIEX products.

If the preamble fails for any reason, a fatal error is reported and the test ends. If it succeeds, i2diag checks to see whether it must download a *.LOD module. If so, it tries to open the appropriate file and write the loadware to the board. If this succeeds, the test proper may begin.

Unlike earlier versions of i2diag, the standard loadware (FF.LOD) is loaded in the same way as other test loadware. So any errors are handled the same for each.

If the preamble is successful, the low-level test first attempts to write the appropriate *.LOD module. If this succeeds, the test proceeds as described in the sections below. The following section shows the error messages which may occur if the preamble or download does not succeed.

Preamble Errors

The following error messages describe problems with resetting and initializing the board, writing loadware to it, and other operations common to most tests.

Board Initialization Errors:

They all begin with the message:

`Error Initializing board at address 0x`*`NNN`*

(Where NNN is the base address, in hexadecimal, where the board is expected).

Then one of the following messages will show the cause of the trouble:

`Proposed board address is invalid.`

This means you have specified a board address (using the a argument) which is impossible: Too big or small, or not a multiple of 8.

`Power-On Reset Message did not begin properly: `*`N`*** **`characters received.`**

(Where N is the number of characters actually read in.) The Power-On Reset Message must begin with two particular bytes. If it does not, we suspect we might not be addressing an IntelliPort-II or -IIEX board, and quit immediately. The bytes actually read are displayed on the next line.

`Power-On Reset Message too short: `*`N`*** **`characters received.`**

(Where N is the number of characters actually read.) How could this happen? Perhaps an otherwise good board died in the midst of writing the Power-On Reset Message. Or perhaps the firmware is writing the message much too slowly.

`Power-On Reset Message too long: `*`N`*** **`characters received.`**

This actually means that the firmware detected characters in the data FIFO even after the entire Power-On Reset Message was supposed to be read in. How could this be? Perhaps the FIFO's character-present flag is stuck on.

`Power-On Reset Message reports an unsupported board.`

Perhaps this actually some new board that this version of i2diag doesn't "understand" yet. If not, suspect that the host interface FIFO is unreliable.

`Power-On Reset Message has conflicting information.`

This would include things like reporting eight ports present on an ISA-4. The most likely cause would be a sick FIFO.

`Power-on Reset Message reports P.O.S.T error.`

The firmware's self tests reported some error, like bad or missing UART's.

`Expandable board expected, non-expandable found.` (or)

`Non-expandable board expected, expandable found.`

Some tests can only be run on IntelliPort-IIEX (expandable) boards, and others only on IntelliPort-II (non-expandable) boards. The Power-On Reset Message indicates the wrong type of board, based on the tests that were chosen, or based on possible EX or EN command line arguments.

Errors while Writing Loadware

After the board is initialized, errors can be encountered when trying to write any of the loadware modules. If the name of the loadware file being used hasn't already displayed, you will see the message:

`Error loading` *`aaaaa`*

(Where aaaaa is the name of the file containing the loadware).

Then one of these explanations appears:

`CRC error on download`

A valid loadware file always contains a CRC in the header to guard against corruption during download or against corruption of the loadware file itself. The bootstrap firmware generates a CRC as well and compares it with the CRC sent down in the header. If they do not match, the firmware sends a "bad acknowledgment" back to the host.

A bad CRC is either because the file itself is corrupt, or because there is something unreliable in the host interface.

`Bad magic number in the loadware file`

This almost certainly means your file is corrupt. I2diag hasn't even tried to send it to the board.

`Loadware File too large` *`(and)`* **`Loadware File too small`**

These mean that the actual size of the file is larger or smaller than the size reported in the file's header. Again, this suggests that the file itself is corrupt.

`Cannot Open File`

Almost certainly because it isn't there. Either you failed to install all the loadware files you need, or you have deleted some.

High-Level Preamble Errors

When the standard loadware (FF.LOD) is written, a number of other initialization routines are run, to install the interrupt handler, and initialize interface structures.

If any of these find errors, you will again see the **`Error initializing board...`** message described above. The following conditions can occur when initializing the standard loadware:

`Power-On Reset Message has a discrepancy between actual ports found and what this product should support.`

For example, an IntelliPort-IIEX shows that there are three 16-port boxes installed, but the actual map of UART's found shows a box having only three (!) UART's. If this error happens, both channel-maps are displayed to highlight the difference.

`Unsupported IRQ selected.`

Probably only could occur from a bad command-line argument (iNN-NN), going undetected. It means that we tried to initialize the interrupt handler for an interrupt that we know this board is incapable of generating on any system.

Mere interrupt conflicts would not be reported in this way.

`No Channel Structures have been Initialized yet.`

Shouldn’t actually happen. This is a software sanity check.

`Channels specified but not reported present.`

If the C argument is used, any channel selected for test must be present on the board. If not, the high-level test will fail at the preamble with this message.

T1 Fifo Sanity Check

Default passes: 1

Maximum passes: 1

Description: Host validates FIFO registers

Errors (all fatal): *Preamble errors: see above*
Initial AF/AE register error
Initial mask register error
AF/AE register test failed
Mask register test failed
Byte-detect register test failed
Outbound mailbox test failed

Loadware: None

Required command line arguments:

EN

Restrictions: Only for IntelliPort-II controllers.

Summary:

This test is valid only for (original, non-ASIC, non-expandable) IntelliPort-II boards, since it tests registers which are peculiar to the AMD biFifo used on these products. The EN command line argument is required as a "promise" that this will be not be an IntelliPort-IIEX board, since it does not try to read the Power-On Reset Message.

After the preamble, the host verifies it can write, read back, and compare random data values using some biFifo internal registers. If data does not match, the value written/read is displayed.

This is an "easy" test: failure usually means bad board addressing. Check DIP switch.

T2 Power-On Reset Message Test

Default passes: 1

Maximum passes: 1

Description: Host displays the board's Power-On Reset Message

Errors: *Preamble Errors: see above*

Loadware: None

Required command line arguments:

None

Restrictions: None

Summary:

After the preamble, this "test" merely displays the Power-On Reset Message. Then it displays the information it contains, all decoded so that it makes sense. Then the operator is prompted to hit the return key, before it continues with the next test.

This test is useful in case other tests have been complaining about the Power-On Reset Message. You can concentrate on just the one thing.

T4 Interactive LCD Controller Test

Default passes: 1

Maximum passes: 1

Description: Checks out LCD controller, keys, and display

Errors: *Preamble Errors: see above*

Loadware: REGL.LOD

Required command line arguments:

None

Restrictions: Only for IntelliPort-IIEX
Boxes with LCD displays must be attached.
Requires operator intervention.

Summary:

After the preamble, the test ensures that the selected board is an IntelliPort-IIEX controller. If not, it reports that the wrong type of board was found for this test. Otherwise the loadware module is written to the board.

The loadware turns the LED green, then sets up a link between the host interface and LCD controller interface. Data coming in through the host mailbox register is used to select which box's LCD to test. Then any data coming from the host through the FIFO data port is sent to the LCD controller, and any data from the LCD controller is sent to the host.

The host assumes that every box reported as present by the Power-On Reset Message will have an LCD controller. It tests them in order, starting with the box closest the controller.

On each box, the LCD unit will display prompts, and you must do what it says for the test to proceed. You are asked to depress and hold a certain key, then to release it. This is to confirm that all the keys work.

Then it displays a string of characters slowly from left to right. As it displays them, the entire screen should change brightness, from the brightest to the faintest possible values. You must observe this and note whether there actually is a brightness difference. If not, you must flag the box for further study.

Lastly, the version number of the LCD microcontroller firmware is displayed, along with whether this is an 8-port or 16-port box.

Then the next box is tested; so on for each box.

This test differs from most others because it is the test itself does not verify anything. If the switches are faulty, the test may hang forever waiting for you to press them. Or if the display is bad, you won't know that you must press anything. But the test will never actually tell you what is wrong.

If the LCD displays garbage instead of the expected prompts, or the test doesn't notice you've hit the keys, or proceeds even though you haven't hit any keys (!), or does anything unexpected, you must note this and fail the box. The test, if it completes, will not report any errors.

T6 Fifo Measurement Test

Default passes: 1

Maximum passes: 1

Description: Host directly measures the size of the FIFO.

Errors: **`Output FIFO will not drain`**
`FIFO size not as reported`

Loadware: NULL.LOD

Required command line options:

None

Restrictions: None

Summary:

After the preamble, the loadware module is written to the board. It does nothing except blink the LED between green and red very fast. So it will look yellow to mortals. In particular, it won't try to read any data from the host data FIFO.

The test first makes sure the outgoing FIFO is empty. If it does not become empty soon, the test fails, reporting **`Output FIFO will not drain`**.

Then it writes data to the FIFO until the full-flag becomes set. The amount of data written is the size of the FIFO. This number is compared to the size reported in the Power-On Reset Message. If they are different, the test fails, reporting **`FIFO size not as reported`**.

If successful, it reports the FIFO size.

T7 External Bus Exerciser Test

Default passes: 2 Seconds

Maximum passes: 30000 Seconds

Description: Writes and reads devices on the external bus.

Errors (all fatal): *Preamble Errors: see above*

Loadware: REGT3.LOD

Required command line arguments:

None

Restrictions: None

Summary:

This test is provided to help the technician troubleshoot a board which has failed the Board/Box hardware test (T16). Much of this test addresses devices which lie on the External Bus. On IntelliPort-IIEX boards, this is the bus which runs between the controller board and the external boxes. On standard IntelliPort-II, the bus is still distinct, but is local to the controller card.

After preamble, the loadware is written to the board. It turns the LED green, and without further ado begins to write and read all the UART's (CD1400's) and interrupt controllers (8259's) on all the boxes attached. There is no further contact with the host. The test runs continuously until the board is reset by the host in preparation for the next test.

The continuous activity that results on the external bus allows the technician to detect trace faults and other flaws.

The host will wait for the specified time, then proceed with the next test.

The exact sequence (for CD1400-based expandable products) is:

Choose the next bit pattern from the list: 5A, B4, 69, D2, A5, 4B, 96, 2D

For each box:

Select the interrupt controller by writing to the chip mapper (local I/O address FEh).

Write the value to the interrupt controller twice (local I/O address 02). Then read the interrupt controller twice…

For each UART (CD1400) on each box:

Select the UART by writing to the chip mapper.

Write the value to the UART twice (local I/O address 80h), then read the UART twice.

For standard controllers with CD1400 UARTS, the sequence is the same except there are no interrupt controller chips, and there is only one "box", actually internal to the board.

For IntelliPort Plus controllers, the general method is the same, except that the ST654 scratch registers are used, and the data pattern is always 55H.

T8 Debug-port Register Test

Summary:

This test is currently for internal engineering use only. Additional information may be provided later.

T10 Fifo Mailbox Test

Default passes:	40
Maximum passes:	30000
Description:	Host validates FIFO mailbox
Errors (all fatal):	*Preamble Errors: see above* **`Timeout receiving mailbox data`** **`Timeout sending mailbox data`** **`Host Received bad mailbox data`**
Loadware:	FIFOM.LOD

Required command line arguments:

None

Restrictions: None

Summary:

After preamble, the loadware module is written to the board. The loadware turns the LED green and waits for mailbox data from the host. Then it writes it back, toggles the LED to yellow, and waits for more mailbox data. The LED toggles between yellow and green too quickly for the eye to follow.

The host writes test data to the output mailbox, and waits to receive it in the input mailbox. Errors are returned if the incoming data does not match, or if the loadware fails to read or write the mailbox.

T11 Interrupt Test

Default passes:	1000
Maximum passes:	15000
Description:	Board generates selected interrupts to host
Errors:	*Preamble Errors: see above* **`Interrupt Test Failed`** **`Missing Interrupts`** **`Extra Interrupts`** **`Extra and Missing Interrupts`**
Loadware:	INTT.LOD

Required command line arguments:

`I`	To specify interrupt or interrupts to test (non-PCI only).
`BM`	If testing Micro Channel cards.
`BE`	If testing EISA cards
`BP`	If testing PCI cards

Restrictions: None

Summary:

After preamble, the loadware is written to the board. It turns the LED green, then waits for the host to write an interrupt number through the FIFO data port. As soon as the selection is received, the board selects this interrupt line to generate an interrupt (at the same time toggling the LED between green and yellow), and then generates the interrupt by writing to the output mailbox. Then it waits for the next interrupt number from the host, and the process repeats.

non-PCI Controllers:

The host patches interrupt handlers into all the supported IRQ's (currently 3, 4, 5, 7, 10, 11, 12, and 15) regardless of the interrupts selected for test. On receipt of any interrupt on these lines, the handler increments an interrupt count for that line and satisfies the condition by reading the mailbox.

For each pass, the host sends a byte requesting an interrupt on each selected line. After waiting sufficient time for the board to respond, it compares the count of interrupts received with those requested. Discrepancies are listed as "missing" or "extra" IRQ's.

If you are testing a Micro Channel or EISA board, you must use the BM or BE command-line argument, respectively. The interrupt test will then use the board's POS or configuration registers to reconfigure the board for each selected interrupt, before requesting the interrupt from each board. When all passes are complete, the original configuration is restored.

PCI Controllers:

PCI controllers are tested in the same way as non-PCI controllers, except that it can only test the interrupt that the host's PCI BIOS has configured the card to use. The I argument is not required, but if present the test will validate whether this is the same as the one selected by the PCI BIOS.

It is not necessary to test the PCI controllers using multiple interrupts, since the board drives the same signals regardless of the interrupt selected. It is the host's responsibility to map this interrupt request line into the selected interrupt.

Diagnostic Output

See also the O (Output) command line option. If you so specify, you can get a table showing the number of interrupts issued (or at least requested from the board) versus the number caught by the interrupt handler. This is useful in case there are errors, to know just how bad things are.

T12 Random-Pattern Board Memory Test

Default passes: 5

Maximum passes: 30000

Description: Board performs random-pattern DRAM test

Errors (all fatal): *Preamble Errors: see above*
`Invalid on-board memory size`
`On-board memory test failed`
`Timeout while starting test`
`Timeout waiting for pass to complete`

Loadware: MEMR.LOD

Required command line arguments:

None

Restrictions: None

Summary:

After preamble, the loadware module is written to the board. The loadware turns the LED green and waits for two bytes from the host through the FIFO data port: the number of passes. Then it returns a single byte which indicates the amount of DRAM found. The loadware then performs a random-pattern diagnostic. As each pass of memory is complete, a zero is written through the FIFO data port to the host (to indicate progress), and the LED is toggled between green and yellow. This will be slow enough to follow with the eye.

If an error is found, the address, data written, and data read, are sent back to the host in the FIFO. The LED is turned solid red.

The host, after writing the number of passes, checks for the reported memory size, reporting an error if it is invalid or missing. It then monitors the FIFO for progress, and reports if passes do not occur as quickly as expected. If errors are reported through the FIFO, they are displayed, and the test ends.

Technical note: Both this test and the walking-bit memory test execute from DRAM, since they are down-loaded. This creates exposure in two ways.

First, if the DRAM is sufficiently bad, the test will fail to run at all, or may “blow up” in unexpected ways. Should this occur, we would expect to get the error “Timeout while waiting for pass to complete”, since it is not likely to be writing the zeros for each pass, as expected. Of course, if the DRAM subsystem is very bad, we may also fail the board’s power-on self-test. (Fatal error).

Second, the test code does not presently re-locate itself to allow explicit testing of the DRAM from which it is executing. This amounts to about 1k of memory.

The “random” patterns are generated by first selecting an initial value V and an increment I. Successive words in each block under test are set to V, V+I, V+2I, and so on (modulo 65536). An entire block is written, then read back. Different initial and incremental values are used each pass.

This type of test is good at detecting pattern-dependent failures internal to the DRAM chip. This is because, unlike the walking-bit test, the test data varies from one pass to the next. Over a long time, more different patterns will obtain.

T13 Fifo Bi-Directional Loopback Test

Default passes: 250

Maximum passes: 30000

Description: Host writes, reads FIFO data port
Both the board and host check integrity

Errors: *Preamble Errors: see above*
`Data Mismatch Error`
{**`High Byte`**|**`Low Byte`**|**`High & Low Bytes`**}
{**`Writing to board`**|**`Reading from board`**}
`Wrote data` *xxxx*, **`Read back`** *xxxx*

Loadware: FIFOI.LOD

Required command line arguments:

None

Restrictions: None

Summary:

After the preamble, the loadware is written to the board. It turns the LED green, then determines if the board is configured for 16-bit data transfers to the host. If so, it will attempt to use the entire 16-bit path when looping back data. Otherwise, it will just loop back bytes.

For each pass, the loadware reads a sequence of data from the host, writing each word (or byte) back to the host as soon as it is received. The loadware knows what this sequence is supposed to be, and so it checks the integrity of the data it receives. If it detects an error, it writes an error code to the mailbox, turns the LED red, and halts. When 16-bit transfers are used, this code also indicates whether there was a discrepancy in the low byte, high byte, or both. If the pass is entirely correct, it writes a “pass” code to the mailbox, and waits a new pass of data.

With each pass, the loadware toggles the LED green, yellow, green, yellow...

The host, then, sends data to the board through the FIFO data port. It uses 16-bit transfers if the board is so configured, 8-bit otherwise. It sends a pattern according to a formula known to the board. After writing 256 words (or 512 bytes) it tries to read the data back which the board will have looped back. If there is a mis-match, it reads the mailbox to determine whether the board realized the error.

For IntelliPort-IIEX boards, this allows you to determine which of the four FIFO chips may be faulty. One pair of FIFO’s (low byte and high byte) handle data from the host to the board. If one of them is bad, the board should have noticed the error. The other pair of FIFO’s (low and high byte) handle data from the board to the host. If these only are bad, the board will not have noticed any error.

If 16-bit transfers are used, the byte containing the discrepancy tells which FIFO of the pair is bad. For 8- bit transfers, only the low byte is used.

T14 Host Interface Timing Stress FIFO Loopback Test

Default passes:	200
Maximum passes:	30000
Description:	Host validates FIFO data port
Errors (all fatal):	*Preamble Errors: see above* **`Timeout getting data`** **`Data mismatch error`** **`wrote data`** *xxxx*, **`read back`** *xxxx*
Loadware:	FIFOL.LOD

Required command line arguments:

None

Restrictions: None

Summary:

After the preamble, the loadware is written to the board. This loadware turns the LED green and waits for FIFO data from the host. As soon as data is available, the board reads it and writes it back to the host. The board always performs 16-bit reads and writes here: if the host is only using an 8-bit interface, the upper byte is garbage anyway. The LED remains green.

The host will perform its test using words if the board is configured for 16-bit operation, byte otherwise. In the paragraph that follows, we call then “units”.

For each pass, the host writes and reads back data from the FIFO in two ways; first, by writing and reading back single units, then, by writing and reading groups of eight units. This is done to cause a variety of host-bus access timings. An error is reported if the data isn’t getting copied back by the loadware, or if the data read back doesn’t match what was sent.

T15 FIFO Block-Message Loopback test

Default passes:	300
Maximum passes:	30000
Description:	Host writes, reads, checks blocks of data
Errors (all fatal):	*Preamble Errors: see above* **`Timeout getting data`** **`Data mismatch error`** {**`High Byte`**|**`Low Byte`**|**`High & Low Bytes`**} **`wrote data`** *xxxx,* **`read back`** *xxxx*
Loadware:	FIFOB.LOD

Required command line arguments:

None

Related command line arguments:

+*nnnn*	Specify numerical value of data to use for test. (Otherwise random).

Restrictions: None

Summary:

This test is intended to reveal any internal problems with the FIFO chip(s). It uses string transfer instructions to read and write variable-sized blocks of data through the FIFO data port, using the mailbox registers to indicate that blocks have been completely written or read. This test most closely resembles the manner in which data is moved between board and host when the standard loadware is in use (high level tests), except that interrupts are not used, either on board or by the host.

If the controller has been configured for a 16-bit host interface, then word transfers will always be done. Otherwise, byte transfers will be used.

After preamble, the module FIFOB.LOD is written. This loadware first turns the LED green. Each pass begins with the board awaiting mailbox data from the host.

The host begins a pass by writing a variable-length block of random data from RAM to the FIFO data port. Blocks used in this test may range from 8 to 256 characters, and start with two bytes (or a word) which indicate the number of characters remaining in the block. The length is written using OUTB's (or OUTW), the data proper using REP OUTSB (or REP OUTSW). When the host has written the entire block it writes to the output mailbox. It then awaits mailbox data from the board.

Having received mailbox data, the board now reads the first block from the host. Then it sends mailbox data to the host to indicate it has read the first block. It then immediately writes the data to the host through the FIFO data port, and (as soon as the host has read the previous mailbox data) sends another mailbox datum to the host. It then awaits another mailbox datum from the host.

When the host receives the first mailbox datum from the board, it writes a second variable-length block of data, and sends its second mailbox datum to the board. (The board can proceed to the following paragraph while the host is finishing up this one.) Then the host awaits the second mailbox datum from the board (sent in the preceding paragraph). This indicates that the first block of data has been echoed back and is ready in the FIFO. The host reads it back and compares it to what was sent. Any mis-matching characters are reported as an error. If there are no errors, the host awaits a third mailbox datum from the board.

Meanwhile the board, having received the second mailbox datum from the host, reads the second data block and immediately echoes it back to the host through the FIFO data port. When this is complete, it issues the final mailbox datum to the host. From the board's point of view, this concludes a single pass. The board toggles the LED between green and yellow after each pass, and starts again, waiting for the host's first mailbox datum.

Finally, the host reads back the second data block, compares it, and reports any errors. If there were no errors and more passes are needed, the host picks two different block lengths, generates different random data, and repeats the above process.

This test algorithm allows the host and board to be both writing the FIFO at the same time, and at other times to be both reading the FIFO. By testing two separate blocks of data, the host can be comparing the first block while the board is still echoing the second.

T16 Board/Box Hardware Register Test

Default passes:	5
Maximum passes:	30000
Description:	Board checks hardware register reliability
Errors (all fatal):	*Preamble Errors: see above* **`Board hardware register test failed`** **`Unexpected error encountered`**
Loadware:	REGT.LOD

Required command line arguments:

None

Restrictions: None

Summary:

After preamble, the loadware is written to the board. It turns the LED green and waits for the host to supply the number of passes to perform (through the FIFO). It then writes and reads back various hardware registers. If a data mismatch is detected, it reported to the host through the FIFO, and the LED is turned red. Otherwise, at the completion of each pass the LED is toggled between green and yellow, and a zero is written to the FIFO to indicate progress.

On an IntelliPort-II controller, some internal CPU peripheral registers are tested. Then the chip-mapper is tested, and finally any UART's on the board.

On an IntelliPort-IIEX controller, some internal CPU peripheral registers are tested. Then, for each expansion box that is attached, the chip mapper, interrupt controller, all UART's, and the LCD controller (if present) are tested.

To test the LCD controller, special loopback commands are sent. This validates the LCD controller interface to the local CPU but of course does not confirm that the display or push-buttons are working correctly.

This test can be run on an IntelliPort-IIEX without any boxes attached, but not much will get tested.

The host, after writing the number of passes, monitors the FIFO for progress. If a failure is reported through the FIFO, it displays the offending chip, address, and data written and read. If progress information does not appear as quickly as expected or if the error information in the FIFO is not complete, the host complains of an **`unexpected error`**.

T18 FIFO Loopback Using DMA

Default passes: 15

Maximum passes: 30000

Description: Board writes and reads FIFO using DMA.

Errors (all fatal): *Preamble Errors: see above*
`Host timed out reading/writing mailbox`
`Host timed out waiting for FIFO data`
`Board timed out sending data to host`
`Board timed out reading data from host`
`Board received mis-matched data`

Loadware: FIFOD.LOD

Required command line arguments:

None

Restrictions: Can only be run on IntelliPort-IIEX controllers.

Summary:

After the preamble, the loadware is written to the board. It first turns the LED green. Then it initializes a block of 16K bytes of test data. It then waits to receive something from the host mailbox. When it does, it means that the host is ready to begin a pass of the test. The loadware sets up two DMA transfers, one to send the test data to the host through the FIFO, and a second one to read data back from the host (as it becomes available) into a buffer in RAM.

The loadware waits until both transfers are complete. If this seems to take too long, the board will time out, and send a message to the host through the mailbox. After the transfers are complete, it compares the data in the two buffers, and reports back to the host, using the mailbox, whether there were any errors.

This is the end of a pass. The loadware toggles the LED between yellow and green after each pass. Then it waits for another byte from the mailbox to indicate the host wants another pass.

We are deliberately transferring more than a FIFO's worth of data in each pass. Under these circumstances, the board is almost guaranteed to run ahead of the host (which will be looping the data back using programmed i/o), filling up the host-bound FIFO. Similarly, the in-bound FIFO will not be filled as quickly as the board is prepared to empty it. Now the FIFO's empty/full flags are what issue the DMA requests. These must work, or else we are sure to lose data.

At the beginning of each pass, the host writes something (anything!) to the mailbox. Then it waits up to 100 mS for data to start coming in. Then it reads incoming FIFO data and writes it back to the outbound FIFO, 16K times. If ever the inbound FIFO is empty when we expect data, it reports an error (**`Host timed out waiting for FIFO data`**). After each pass, it reads the inbound mailbox to determine whether the pass was successful. If successful, it proceeds to the next pass. If not, one of the above errors are reported.

T19 Walking-Bit Board Memory Test

Default passes: 1

Maximum passes: 30000

Description: Board performs walking-bit DRAM test

Errors (all fatal): *Preamble Errors: see above*
`Invalid on-board memory size`
`On-board memory test failed`
`Timeout while starting test`
`Timeout waiting for pass to complete`

Loadware: MEMW.LOD

Required command line arguments:

None

Restrictions: None

Summary:

After preamble, the loadware module is written to the board. The loadware turns the LED green and waits for two bytes from the host through the FIFO data port: the number of passes. Then it returns a single byte which indicates the amount of DRAM found. The loadware then performs a walking-bit diagnostic. For this purpose, memory is divided into 64k blocks. After each block is tested, the LED it toggled between yellow and green. As each pass of memory is complete, a zero is written back out the FIFO to the host (to indicate progress), and the test block size is cut in half (256-byte minimum). When performing multiple passes, this causes the LED to flash faster and faster for each of the first 9 passes.

If an error is found, the address, data written, and data read, are sent back to the host in the FIFO. The LED is turned solid red.

The host, after writing the number of passes, checks for the reported memory size, reporting an error if it is invalid or missing. It then monitors the FIFO for progress, and reports if passes do not occur as quickly as expected. If errors are reported through the FIFO, they are displayed, and the test ends.

Technical note: Both this test and the random-pattern memory test execute from DRAM, since they are down-loaded. This creates exposure in two ways.

First, if the DRAM is sufficiently bad, the test will fail to run at all, or may “blow up” in unexpected ways. Should this occur, we would expect to get the error **`Timeout while waiting for pass to complete`**, since it is not likely to be writing the zeros for each pass, as expected. Of course, if the DRAM subsystem is very bad, we may also fail the board’s power-on self-test. (Fatal error).

Second, the test code does not presently re-locate itself to allow explicit testing of the DRAM from which it is executing. This amounts to about 1k of memory.

Walking Bit Test Algorithm:

- For each block under test:
  - Write each word to 0.
  - Read each word, verify it is 0.
  - For each word in the block:
    - Read and verify it is 0.
    - Write and read back the following values:
      - 0x0001, 0x0002, 0x0004, 0x0008
      - 0x0010, 0x0020, 0x0040, 0x0080
      - 0x0100, 0x0200, 0x0400, 0x0800
      - 0x1000, 0x2000, 0x4000, 0xc8000
    - Write 0xffff
  - Read back each word, verify 0xffff

T23 AMCC Operation Register Checkout Test

Default passes: 5

Maximum passes: 30000

Description: Host writes/reads board's AMCC interface registers

Errors (fatal): **`AMCC Error (OMB)`**
`Address` *xxxx* **`Wrote`** *xx*, **`Read`** *xx*
`This test only for PCI (use BP argument)`

Loadware: None

Required command line arguments:

`BP`

Restrictions: Can only be run on PCI boards.

Summary:

This test does not use any loadware, nor does it even look at the board through the mailbox or FIFO. The command line argument BP is required as a promise that we are testing a PCI board.

Because of the BP argument, i2diag has already determined that the board is indeed a PCI board, and found its slot. Had this failed, it would have issued an error earlier before any of the tests proper would have run (see also the discussion under the B option write-up).

The test writes and reads back data from the PCI interface chip, failing if there is any discrepancy.

T24 POS Register Checkout Test

Default passes:	5
Maximum passes:	30000
Description:	Host writes/reads board's POS registers
Errors (fatal):	**`Micro Channel POS[2] error`** **`Micro Channel POS[3] error`** **`This test only for Micro Channel (use BM argument)`**
Loadware:	None

Required command line arguments:

`BM`

Restrictions: Can only be run on Micro Channel boards.

Summary:

This test does not use any loadware, nor does it even look at the board through the mailbox or FIFO. The command line argument BM is required as a promise that we are testing a Micro Channel board.

Because of the BM argument, i2diag has already determined that the board is indeed a Micro Channel board, and found its slot. Had this failed, it would have issued an error earlier before any of the tests proper would have run (see also the discussion under the B option write-up).

Using the information obtained earlier, it saves the configurations currently saved in the board's POS[2] and POS[3] registers. These registers are normally used to configure the board's I/O address and interrupt. Then assorted values are written and read back from each register. If any of the values do not match, then **`Micro Channel POS[2]`** *(or POS[3])* **`error`** is issued, and the offending values are displayed.

When all passes have been completed, the original configurations are restored.

Note that interrupts are kept turned off during all passes of this test. This is because we have put the Micro Channel bus into setup mode to access the POS registers, which no interrupt handler would be expecting. This means that you cannot abort this test using ctrl-C or ctrl-break.

Normally this will not be a problem, since no malfunction should make the test potentially hang forever. But be careful of specifying an extremely large number of passes.

T25 Internal Data Loopback Test

Default passes:	3
Maximum passes:	30000
Description:	Places UART in internal data loopback mode, sends data, receives, compares.
Errors (fatal):	High-level preamble errors (see above) Failure to specify the interrupt (i option) Timeout sending a command
Errors (**ON**):	**ch** *nn* **pass** *mm* **- no xmit** **ch** *nn* **pass** *mm* **- missing** **ch** *nn* **pass** *mm* **- bad** (These errors are displayed at end of pass.)
Errors (**OE**):	**Error transmitting on channel** *nn*: **sent** *mm* **out of** *kk* **characters.** **Data missing/late on channel** *nn*: **sent** *mm*: *xxxxxxxxxxxxxxxxx* **got** *rr*: *yyyyyyyyyyyyyyyy* **Data mismatch on channel** *nn*: **sent** *mm*: *xxxxxxxxxxxxxxxxx* **got** *rr*: *yyyyyyyyyyyyyyyy* (These are displayed as they occur).
Loadware:	FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non-PCI cards only).

Other associated arguments:

C*n…*	specified channels to test
MM or **M**	affects error reporting
O*xxx…*	affects error messages
+*sssss…*	Specify test data string (otherwise random).
Restrictions:	IntelliPort-IIEX needs boxes attached.

Summary:

After the standard loadware has been written and initialized, all selected UART channels are placed in internal loopback mode.

For each few passes, commands are sent to each selected channel to set the baud rate, number of stop bits, parity, etc., either randomly or in agreement with any hyphen (-) options given on the command line. If options are given for baud rate, number of stop bits, parity, character size, and flow control, there is no random element left In this case, these commands are issued just once, before the first pass.

Ordinarily, nothing should prevent these commands from reaching the board and being performed. Should the unexpected happen, and there is not enough room to buffer a command within a reasonable time, the message Timeout sending a command is displayed, and testing ends. Check to see whether the board LED is flashing a

fatal error (see Appendix A). This may indicate a FIFO or DRAM problem untaught by low-level tests. Failure to receive interrupts from the board can also cause this. Make sure there are no interrupt conflicts with other hardware on the host machine.

Then, a string of quasi-random data is sent to each channel. (When eight-bit protocol is selected, this test data will include characters with the eighth bit set). As with the commands, nothing should prevent this data from being sent to the board within a reasonable time. If this does not occur, the message “Error Transmitting...” or “...-no xmt” is displayed. Unlike the time-out on sending a command, this error is not immediately fatal. Instead, this channel will be omitted from further passes of this test.

While a transmit error is not fatal, the probable causes are the same as with the command time-outs, and the remaining channels are likely to fail soon as well. Once all selected channels have failed, no more passes of that test are run.

Once the data is sent, we wait a reasonable time for the data to be received. Since this is an internal loopback test, external loopback plugs are not necessary. In fact, in internal loopback mode, the output data is not sent to the UART Tx pin. A breakout box connected to the channel will show no activity.

If the data is not all received within a reasonable time, the message “Data missing/late...” or “...-missing” is displayed. The channel is omitted on future passes of this test. If the data is all received, it is compared with what was sent. If it does not match, the message “Data mismatch...” or “...-bad” is displayed. The channel is omitted on future passes of this test.

Technician’s Hint: The internal loopback test should always be run when an external data loopback test (see below) reports failures. If the internal loopback test passes, it suggests that the failure is in the path between the UART’s, the RS232 drivers or receivers, the connectors, and the loopback plug. If the internal loopback test fails as well, it suggests the failure is either internal to the UART’s, or in the data bus between these chips and the processor itself.

It will be easier to determine the cause of failures if you use the output control OEN. If data is missing, you can determine how much, if any, came through. If data is bad, you can compare the strings to see how much data was corrupted, and in what way.

If fatal errors are reported by the board LED, or several channels fail in the same way, it suggests there may be a more basic problem than UART’s. Try re-running increased passes of all the low-level diagnostics to flush it out.

T26 External Data Loopback Test

Default passes: 10

Maximum passes: 30000

Description: Sends data to UART: data is looped back through an external plug, received, compared.

Errors (fatal): High-level preamble errors (see above)
Failure to specify the interrupt (i option)
Timeout sending a command

Errors (**ON**):
ch *nn* **pass** *mm* **- no xmit**
ch *nn* **pass** *mm* **- missing**
ch *nn* **pass** *mm* **- bad**
(These errors are displayed at end of pass.)

Errors (**OE**):
Error transmitting on channel *nn*:
sent *mm* **out of** *kk* **characters.**
Data missing/late on channel *nn*:
sent *mm*: *xxxxxxxxxxxxxxxxx*
got *rr*: *yyyyyyyyyyyyyyyy*
Data mismatch on channel *nn*:
sent *mm*: *xxxxxxxxxxxxxxxxx*
got *rr*: *yyyyyyyyyyyyyyyy*
(These are displayed as they occur).

Loadware: FF.LOD (standard)

Required command line arguments:

I to specify the interrupt (non-PCI cards only).

Other associated arguments:

C*n…* specified channels to test

MM or **M** affects error reporting

O*xxx…* affects error messages

+*sssss…* Specify test data string (otherwise random).

Restrictions: IntelliPort-IIEX needs boxes attached
Loopbacks with at least Tx-Rx.

Summary:

After the standard loadware has been written and initialized, all selected UART channels are placed in normal mode.

This test is completely identical to the T25, the internal data loopback test, except that the UART's are running in normal mode, rather than internal loopback mode. During testing, RTS and DTR are asserted.

External loopback plugs must be attached to every channel under test. At a minimum, these plugs must connect TxD to RxD. Because of the requirements for other tests, the plugs generally loop back DCD-DTR and CTS-RTS as well.

See the description under T25 for an outline of the test process and an interpretation of the errors.

T27 External Data Loopback Test (RTS & DTR low)

Default passes: 10

Maximum passes: 30000

Description: Sends data to UART: data is looped back through an external plug, received, compared.

Errors (fatal): High-level preamble errors (see above)
Failure to specify the interrupt (i option)
Timeout sending a command

Errors (**ON**):
ch *nn* **pass** *mm* **- no xmit**
ch *nn* **pass** *mm* **- missing**
ch *nn* **pass** *mm* **- bad**
(These errors are displayed at end of pass.)

Errors (**OE**):
Error transmitting on channel *nn*:
sent *mm* **out of** *kk* **characters.**
Data missing/late on channel *nn*:
sent *mm*: *xxxxxxxxxxxxxxxxx*
got *rr*: *yyyyyyyyyyyyyyyy*
Data mismatch on channel *nn*:
sent *mm*: *xxxxxxxxxxxxxxxxx*
got *rr*: *yyyyyyyyyyyyyyyy*
(These are displayed as they occur).

Loadware: FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non-PCI cards only).

Other associated arguments:

C*n…*	specified channels to test
MM or **M**	affects error reporting
O*xxx…*	affects error messages
+*sssss…*	Specify test data string (otherwise random).

Restrictions: IntelliPort-IIEX needs boxes attached
Loopbacks with at least Tx-Rx.

Summary:

After the standard loadware has been written and initialized, all selected UART channels are placed in normal mode.

This test is completely identical to the T25, the internal data loopback test, except that the UART's are running in normal mode, rather than internal loopback mode. During testing RTS and DTR are negated.

External loopback plugs must be attached to every channel under test. At a minimum, these plugs must connect TxD to RxD. Because of the requirements for other tests, the plugs generally loop back DCD-DTR and CTS-RTS as well.

See the description under T25 for an outline of the test process and an interpretation of the errors.

T28 External Data Loopback with Data Set Exercise Test

Default passes: 10

Maximum passes: 30000

Description:	Sends data to UART: data is looped back through an external plug while DTR and RTS are being toggled; data is received, compared.
Errors (fatal):	High-level preamble errors (see above) Failure to specify the interrupt (i option) Timeout sending a command
Errors (**ON**):	*See "T26 External Data Loopback Test" on page 58.*
Errors (**OE**):	*See "T26 External Data Loopback Test" on page 58.*
Loadware:	FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non-PCI cards only).

Other associated arguments:

C*n…*	specified channels to test
MM or **M**	affects error reporting
O*xxx…*	affects error messages
+*sssss…*	Specify test data string (otherwise random).

Restrictions:	IntelliPort-IIEX needs boxes attached. Loopbacks with at least Tx-Rx.

Summary:

After the standard loadware has been written and initialized, all selected UART channels are placed in normal mode.

This test is completely identical to the T26, the external data loopback test, except that while waiting for the incoming data, bypass commands are continuously sent to all selected channels, to raise and lower DTR and RTS as quickly as possible. (No checking of CTS or DCD is done, however). This additional action is an attempt to induce errors by creating more noise on lines which would be otherwise dormant.

If this test were to fail while the regular external data loopbacks passed, it might indicate some crosstalk between the data and data-set signals, or perhaps an internal UART problem.

External loopback plugs must be attached to every channel under test. At a minimum, these plugs must connect TxD to RxD. Because of the requirements for other tests, the plugs generally loop back DCD-DTR and CTS-RTS as well. For this test, such a plug has the advantage of causing the CTS and DCD lines to move as well.

See the description under T25 for an outline of the test process and an interpretation of the errors.

T30 External Data Loopback (RTS/CTS flow control) Test

Default passes: 20

Maximum passes: 30000

Description:	Sends data to UART: data is looped back through an external plug, using CTS/RTS flow control; data is received, compared.
Errors (fatal):	High-level preamble errors (see above) Failure to specify the interrupt (i option) Timeout sending a command
Errors (**ON**):	*See "T26 External Data Loopback Test" on page 58*.
Errors (**OE**):	*See "T26 External Data Loopback Test" on page 58.*
Loadware:	FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non-PCI cards only).

Other associated arguments:

C*n…*	specified channels to test
MM or **M**	affects error reporting
O*xxx…*	affects error messages
+*sssss…*	Specify test data string (otherwise random).

Restrictions:	IntelliPort-IIEX needs boxes attached. Loopbacks with at least Tx-Rx and CTS-RTS

Summary:

After the standard loadware has been written and initialized, all selected UART channels are placed in normal mode.

This test is completely identical to the T26, the external data loopback test, except that RTS/CTS flow control is used, and if the baud rate is not specified, mostly very fast ones will be chosen.

This validates that CTS and RTS flow control function properly. CTS flow control is supported by the UART on all products. RTS flow is supported by the UART on IntelliPort Plus (ST654) products, and by the loadware on original CD1400-based products.

External loopback plugs must be attached to every channel under test. At a minimum, these plugs must connect TxD to RxD, and RTS to CTS. Because of the requirements for other tests, the plugs generally loop back DCD-DTR as well.

See the description under T25 for an outline of the test process and an interpretation of the errors.

T31 External Data Leak Test

Default passes: 1

Maximum passes: 1

Description:	Sends data to UART: data is *not* intentionally looped back. If any data is received, the test fails.
Errors (fatal):	High-level preamble errors (see above) Failure to specify the interrupt (i option) Timeout sending a command
Errors (**ON**):	**ch** *nn* **pass** *mm* **- LEAKING** (These errors are displayed at end of pass.)
Loadware:	FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non-PCI cards only).

Other associated arguments:

C*n…*	specified channels to test
O*xxx…*	affects error messages

Restrictions:	IntelliPort-IIEX needs boxes attached. **No** loopbacks connecting Tx with Rx.

Summary:

After the standard loadware has been written and initialized, all selected UART channels are placed in normal mode.

Data is sent to the UART. Since it is in normal mode and no loopback connector is attached, we do not expect to receive any data. If we do, it means that there is noise, crosstalk, or even a short between the Tx and Rx lines.

T34 External Data Loopback (DTR/CTS flow control) Test

Default passes: 20

Maximum passes: 30000

Description:	Sends data to UART: data is looped back through an external plug, using DTR/RTS flow control; data is received, compared.
Errors (fatal):	High-level preamble errors (see above) Failure to specify the interrupt (i option) Timeout sending a command
Errors (**ON**):	*See "T26 External Data Loopback Test" on page 58*.
Errors (**OE**):	*See "T26 External Data Loopback Test" on page 58.*
Loadware:	FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non-PCI cards only).

Other associated arguments:

C*n…*	specified channels to test
MM or **M**	affects error reporting
O*xxx…*	affects error messages

Restrictions:	IntelliPort-IIEX needs boxes attached. Loopbacks with at least Tx-Rx and DTR-CTS

Summary:

After the standard loadware has been written and initialized, all selected UART channels are placed in normal mode.

This test is completely identical to the T26, the external data loopback test, except that DTR/CTS flow control is used, and if the baud rate is not specified, mostly very fast ones will be chosen.

This validates that CTS and DTR flow control function properly. CTS flow control is supported by the UART on all products. DTR flow is supported by the UART on original CD1400-based products, and by the loadware on IntelliPort Plus (ST654)

External loopback plugs must be attached to every channel under test. At a minimum, these plugs must connect TxD to RxD, and DTR to CTS. (This is "crossed" from our normal convention for manufacturing loopback plugs which cross RTS-CTS and DTR-DCD).

T35 Break Generation & Detection Test

Default passes:	10
Maximum passes:	30000
Description:	Sends (250 mS) breaks to UART, through external plug, receives, compares
Errors (fatal):	High-level preamble errors (see above) Failure to specify the interrupt (i option) Timeout sending a command through the standard library
Errors, (**ON**):	**ch** *nn* **pass** *mm* **- #breaks** (Displayed at end of pass.)
Errors, (**OE**):	**Break loopback channel** *nn* **at pass** *mm* **1 generated,** *k* **detected** (Displayed as it occurs).
Loadware:	FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non-PCI cards only).

Other associated arguments:

C*n…*	specified channels to test
O*xxx…*	affects error messages

Restrictions:	IntelliPort-IIEX needs boxes attached. Loopbacks with at least Tx-Rx

Summary:

After the standard loadware has been written and initialized, all selected UART channels are placed in normal mode.

As with the data loopback tests, a new set of protocols is selected for each pass. These should not affect the generation of breaks, which should always take about a quarter second for each. It does, however, verify that the UART's can properly detect the breaks at a variety of baud rates and other settings. As elsewhere, the hyphen (-) arguments can be used to force particular baud rates, parity, etc. (see Appendix D).

For each pass, the host sends to each selected channel a series of two breaks separated by a delay of approximately 10 mS. If these two breaks are not detected within a reasonable time, an error message is displayed and this channel is omitted from further passes of this test.

External loopback plugs must be attached to every channel under test. At a minimum, these plugs must connect TxD to RxD. Because of the requirements for other tests, the plugs generally loop back DCD-DTR and CTS-RTS as well.

This test is provided to ensure that break generation and detection is properly supported by the UART chips; a data loopback test alone does not prove this. Aside from an internal UART problem or sick loadware, boards which pass the external loopback tests ought to pass this test as well.

T36 DTR/DCD Loopback Test

Default passes:	10
Maximum passes:	30000
Description:	Ensures DCD tracks DTR changes on each channel.
Errors (fatal):	High-level preamble errors (see above) Failure to specify the interrupt (i option) Timeout sending a command.
Errors, (**ON**):	**ch** *nn* **pass** *mm* **- DCD low** **ch** *nn* **pass** *mm* **- DCD up** (These errors are displayed at end of pass.)
Errors, (**OE**):	**DTR-DCD channel** *nn* **at pass** *mm* **DTR high but DCD low** **DTR-DCD channel** *nn* **at pass** *mm* **DTR low but DCD high** (These are displayed as they occur).
Loadware:	FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non-PCI cards only).

Other associated arguments:

C	specified channels to test
O	affects error messages

Restrictions:	IntelliPort-IIEX needs boxes attached Loopbacks with at least DTR-DCD

Summary:

Since the UART's do not support an internal data-set loopback mode, all data-set loopback tests are external, and require loopback plugs.

After the standard loadware has been written, the UART's are initialized to normal (non-loopback) mode, and to report any data-set signal changes.

For the each pass, the following occurs twice:

RTS is lowered on each selected channel.

DTR is set to a random value on each selected channel and DCD is verified to match. (Initially this "random" value is low for all channels).

DTR is reversed on all selected channels, and DCD verified to follow.

RTS is raised on each selected channel.

DTR is reversed, and DCD verified to follow.

DTR is reversed again, and DCD verified to follow.

A new random pattern is selected for DTR assertions.

If DCD fails to track DTR, an error is logged for that port. Up to two failures will be reported on each port, one for each direction (high when should be low, low when should be high). The data-set outputs will continue to be toggled on ports that fail, but the test will not wait for the input to track.

If at any point we are unable to send a command to raise or lower a data-set line, the message **`Timeout sending a command...`** is displayed and testing ends. This probably indicates a centralized problem, such as bad FIFO or DRAM. See also the comments about this message under T25.

T37 RTS/CTS Loopback Test

Default passes:	10
Maximum passes:	30000
Description:	Ensures CTS tracks RTS changes on each channel.
Errors (fatal):	High-level preamble errors (see above) Failure to specify the interrupt (i option) Timeout sending a command.
Errors, (**ON**):	**ch** *nn* **pass** *mm* **- CTS low** **ch** *nn* **pass** *mm* **- CTS up** (These errors are displayed at end of pass.)
Errors, (**OE**):	**RTS-CTS channel** *nn* **at pass** *mm* **RTS high but CTS low** **RTS-CTS channel** *nn* **at pass** *mm* **RTS low but CTS high** (These are displayed as they occur).
Loadware:	FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non-PCI cards only).

Other associated arguments:

C	specified channels to test
O	affects error messages

Restrictions:	IntelliPort-IIEX needs boxes attached Loopbacks with at least RTS-CTS

Summary:

Since the UART's do not support an internal data-set loopback mode, all data-set loopback tests are external, and require loopback plugs.

After the standard loadware has been written, the UART's are initialized to normal (non-loopback) mode, and to report any data-set signal changes.

For the each pass, the following occurs twice:

DTR is lowered on each selected channel.

RTS is set to a random value on each selected channel and CTS is verified to match. (Initially this "random" value is low for all channels).

RTS is reversed on all selected channels, and CTS verified to follow.

DTR is raised on each selected channel.

RTS is reversed, and CTS verified to follow.

RTS is reversed again, and CTS verified to follow.

A new random pattern is selected for RTS assertions.

If CTS fails to track RTS, an error is logged for that port. Up to two failures will be reported on each port, one for each direction (high when should be low, low when should be high). The data-set outputs will continue to be toggled on ports that fail, but the test will not wait for the input to track.

If at any point we are unable to send a command to raise or lower a data-set line, the message **`Timeout sending a command...`** is displayed and testing ends. This probably indicates a centralized problem, such as bad FIFO or DRAM. See also the comments about this message under T25.

T40 DTR/DSR Loopback Test

Default passes:	10
Maximum passes:	30000
Description:	Ensures DSR tracks DTR changes on each channel.
Errors (fatal):	High-level preamble errors (see above) Failure to specify the interrupt (i option) Timeout sending a command.
Errors, (**ON**):	**ch** *nn* **pass** *mm* **- DSR low** **ch** *nn* **pass** *mm* **- DSR up** (These errors are displayed at end of pass.)
Errors, (**OE**):	**DTR-DSR channel** *nn* **at pass** *mm* **DTR high but DSR low** **DTR-DSR channel** *nn* **at pass** *mm* **DTR low but DSR high** (These are displayed as they occur).
Loadware:	FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non-PCI cards only).

Other associated arguments:

C	specified channels to test
O	affects error messages

Restrictions:	IntelliPort-IIEX needs boxes attached Loopbacks with at least DTR-DSR

Summary:

Since the UART's do not support an internal data-set loopback mode, all data-set loopback tests are external, and require loopback plugs.

After the standard loadware has been written, the UART's are initialized to normal (non-loopback) mode, and to report any data-set signal changes.

For the each pass, the following occurs twice:

RTS is lowered on each selected channel.

DTR is set to a random value on each selected channel and DSR is verified to match. (Initially this "random" value is low for all channels).

DTR is reversed on all selected channels, and DSR verified to follow.

RTS is raised on each selected channel.

DTR is reversed, and DSR verified to follow.

DTR is reversed again, and DSR verified to follow.

A new random pattern is selected for DTR assertions.

If DSR fails to track DTR, an error is logged for that port. Up to two failures will be reported on each port, one for each direction (high when should be low, low when should be high). The data-set outputs will continue to be toggled on ports that fail, but the test will not wait for the input to track.

If at any point we are unable to send a command to raise or lower a data-set line, the message **`Timeout sending a command...`** is displayed and testing ends. This probably indicates a centralized problem, such as bad FIFO or DRAM. See also the comments about this message under T25.

Note that on the standard IntelliPort-II, the DCD pin supplies both DCD and DSR. On the IntelliPort-IIEX boxes using DB25 connectors, DSR comes in on pin 11.

T41 RTS/RI Loopback Test

Default passes:	10
Maximum passes:	30000
Description:	Ensures RI tracks RTS changes on each channel.
Errors (fatal):	High-level preamble errors (see above) Failure to specify the interrupt (i option) Timeout sending a command.
Errors, (**ON**):	**ch** *nn* **pass** *mm* **- RI low** **ch** *nn* **pass** *mm* **- RI up** (These errors are displayed at end of pass.)
Errors, (**OE**):	**RTS-RI channel** *nn* **at pass** *mm* **RTS high but RI low** **RTS-RI channel** *nn* **at pass** *mm* **RTS low but RI high** (These are displayed as they occur).
Loadware:	FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non-PCI cards only).

Other associated arguments:

C	specified channels to test
O	affects error messages

Restrictions:	IntelliPort-IIEX needs boxes attached Loopbacks with at least RTS-RI Can only run on IntelliPort-IIEX DB25 and Power RackPort (IntelliPort-II non-expandables and products with 8-pin RJ54's do not support the RI signal).

Summary:

Since the UART's do not support an internal data-set loopback mode, all data-set loopback tests are external, and require loopback plugs.

After the standard loadware has been written, the UART's are initialized to normal (non-loopback) mode, and to report any data-set signal changes.

For the each pass, the following occurs twice:

DTR is lowered on each selected channel.

RTS is set to a random value on each selected channel and RI is verified to match. (Initially this "random" value is low for all channels).

RTS is reversed on all selected channels, and RI verified to follow.

DTR is raised on each selected channel.

RTS is reversed, and RI verified to follow.

RTS is reversed again, and RI verified to follow.

A new random pattern is selected for RTS assertions.

If RI fails to track RTS, an error is logged for that port. Up to two failures will be reported on each port, one for each direction (high when should be low, low when should be high). The data-set outputs will continue to be toggled on ports that fail, but the test will not wait for the input to track.

If at any point we are unable to send a command to raise or lower a data-set line, the message **`Timeout sending a command...`** is displayed and testing ends. This probably indicates a centralized problem, such as bad FIFO or DRAM. See also the comments about this message under T25.

T43 DTR/CTS Loopback Test

Default passes:	10
Maximum passes:	30000
Description:	Ensures CTS tracks DTR changes on each channel.
Errors (fatal):	High-level preamble errors (see above) Failure to specify the interrupt (i option) Timeout sending a command.
Errors, (**ON**):	**ch** *nn* **pass** *mm* **- CTS low** **ch** *nn* **pass** *mm* **- CTS up** (These errors are displayed at end of pass.)
Errors, (**OE**):	**DTR-CTS channel** *nn* **at pass** *mm* **DTR high but CTS low** **DTR-CTS channel** *nn* **at pass** *mm* **DTR low but CTS high** (These are displayed as they occur).
Loadware:	FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non-PCI cards only).

Other associated arguments:

C	specified channels to test
O	affects error messages

Restrictions:	IntelliPort-IIEX needs boxes attached Loopbacks with at least DTR-CTS

Summary:

Since the UART's do not support an internal data-set loopback mode, all data-set loopback tests are external, and require loopback plugs.

After the standard loadware has been written, the UART's are initialized to normal (non-loopback) mode, and to report any data-set signal changes.

For the each pass, the following occurs twice:

RTS is lowered on each selected channel.

DTR is set to a random value on each selected channel and CTS is verified to match. (Initially this "random" value is low for all channels).

DTR is reversed on all selected channels, and CTS verified to follow.

RTS is raised on each selected channel.

DTR is reversed, and CTS verified to follow.

DTR is reversed again, and CTS verified to follow.

A new random pattern is selected for DTR assertions.

If CTS fails to track DTR, an error is logged for that port. Up to two failures will be reported on each port, one for each direction (high when should be low, low when should be high). The data-set outputs will continue to be toggled on ports that fail, but the test will not wait for the input to track.

If at any point we are unable to send a command to raise or lower a data-set line, the message **`Timeout sending a command...`** is displayed and testing ends. This probably indicates a centralized problem, such as bad FIFO or DRAM. See also the comments about this message under T25.

Connecting DTR to CTS is “crossed” from the normal convention used for loopback plugs here, i.e., DTR-DCD,DSR and RTS-CTS and RI.

T44 RTS/DCD Loopback Test

Default passes:	10
Maximum passes:	30000
Description:	Ensures DCD tracks RTS changes on each channel.
Errors (fatal):	High-level preamble errors (see above) Failure to specify the interrupt (i option) Timeout sending a command.
Errors, (**ON**):	**ch** *nn* **pass** *mm* **- DCD low** **ch** *nn* **pass** *mm* **- DCD up** (These errors are displayed at end of pass.)
Errors, (**OE**):	**RTS-DCD channel** *nn* **at pass** *mm* **RTS high but DCD low** **RTS-DCD channel** *nn* **at pass** *mm* **RTS low but DCD high** (These are displayed as they occur).
Loadware:	FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non-PCI cards only).

Other associated arguments:

C	specified channels to test
O	affects error messages

Restrictions:	IntelliPort-IIEX needs boxes attached Loopbacks with at least RTS-DCD

Summary:

Since the UART's do not support an internal data-set loopback mode, all data-set loopback tests are external, and require loopback plugs.

After the standard loadware has been written, the UART's are initialized to normal (non-loopback) mode, and to report any data-set signal changes.

For the each pass, the following occurs twice:

DTR is lowered on each selected channel.

RTS is set to a random value on each selected channel and DCD is verified to match. (Initially this "random" value is low for all channels).

RTS is reversed on all selected channels, and DCD verified to follow.

DTR is raised on each selected channel.

RTS is reversed, and DCD verified to follow.

RTS is reversed again, and DCD verified to follow.

A new random pattern is selected for RTS assertions.

If DCD fails to track RTS, an error is logged for that port. Up to two failures will be reported on each port, one for each direction (high when should be low, low when should be high). The data-set outputs will continue to be toggled on ports that fail, but the test will not wait for the input to track.

If at any point we are unable to send a command to raise or lower a data-set line, the message **`Timeout sending a command...`** is displayed and testing ends. This probably indicates a centralized problem, such as bad FIFO or DRAM. See also the comments about this message under T25.

Connecting RTS to DCD is “crossed” from the normal convention used for loopback plugs here, i.e., DTR-DCD,DSR and RTS-CTS and RI.

T45 Data Transmission Test

Default passes:	2 seconds
Maximum passes:	30000 seconds
Description:	Sends a string of data continuously to the UART
Errors (fatal):	High-level preamble errors (see above) Failure to specify the interrupt (i option). Timeout sending a command
Loadware:	FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non-PCI cards only)

Other associated arguments:

C	specified channels to test
O	option affects error messages

Restrictions: IntelliPort-IIEX needs boxes attached.

Summary:

After the standard loadware has been written and initialized, all selected UART channels are placed in normal mode.

Then (unless the Y option was given) you are prompted to enter the data string to be sent to all the selected channels.

Then a protocol is chosen for each selected channel. As with other tests, this can be forced using the hyphen (-) options (see Appendix D).

Then the selected data string is transmitted continuously to all the selected channels for a time (in seconds) equal to the number of passes selected.

Except for unexpected errors due to low-level problems, this test always “passes”, because no special checking is performed. The purpose of this test is to aid in signal tracing. If a data loopback test were to fail on this channel, how far is the signal going? Is it getting to the RS232 drivers? The receivers? The UART receive pin? It is permissible to run this test with data loopback plugs connected, provided that xon/xoff flow control is not specified on the command line.

Another use of this test is to verify the correctness of a particular baud rate and protocol, by connecting the output to a terminal. A full set of protocol options should be used in this case.

T46 DTR Tracing Test

Default passes:	2 seconds
Maximum passes:	30000 seconds
Description:	Raises and lowers DTR continuously
Errors (fatal):	High-level preamble errors (see above) Failure to specify the interrupt (i option). Timeout sending a command
Loadware:	FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non-PCI cards only)

Other associated arguments:

C	specified channels to test
O	option affects error messages

Restrictions: IntelliPort-IIEX needs boxes attached.

Summary:

After the standard loadware has been written and initialized, all selected UART channels are placed in normal mode.

Then DTR is raised and lowered continuously on all the selected channels. This continues for a time (in seconds) equal to the number of passes specified. Delays are inserted between each command to raise or lower DTR so that the resulting signal should have the following shape: DTR up for 10 milliseconds, down for 30 milliseconds.

As with the data transmission test, there is no checking performed, so this test always “passes”. The purpose, again, is to facilitate signal tracing.

T47 RTS Tracing Test

Default passes:	2 seconds
Maximum passes:	30000 seconds
Description:	Raises and lowers RTS continuously
Errors (fatal):	High-level preamble errors (see above) Failure to specify the interrupt (i option). Timeout sending a command
Loadware:	FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non PCI cards only)

Other associated arguments:

C	specified channels to test
O	option affects error messages

Restrictions: IntelliPort-IIEX needs boxes attached.

Summary:

After the standard loadware has been written and initialized, all selected UART channels are placed in normal mode.

Then RTS is raised and lowered continuously on all the selected channels. This continues for a time (in seconds) equal to the number of passes specified. Delays are inserted between each command to raise or lower RTS so that the resulting signal should have the following shape: RTS up for 10 milliseconds, down for 20 milliseconds. (Different duty cycle from DTR tracing test).

As with the data transmission test, there is no checking performed, so this test always “passes”. The purpose, again, is to facilitate signal tracing.

T48 Data Echo Test

Default passes:	2 seconds
Maximum passes:	30000 seconds
Description:	Reads and echoes incoming characters.
Errors (fatal):	High-level preamble errors (see above) Failure to specify the interrupt (i option). Timeout sending a command
Loadware:	FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non-PCI cards only)

Other associated arguments:

C	specified channels to test
O	option affects error messages

Restrictions: IntelliPort-IIEX needs boxes attached.

Summary:

After the standard loadware has been written and initialized, all selected UART channels are placed in normal mode.

Then a protocol is chosen for each selected channel. As with other tests, this can be forced using the hyphen (-) options (see Appendix D).

Then the test polls all the selected channels for incoming characters. Any characters received are immediately sent back to the board to be re-transmitted.Note that the that data flows from the UART through the board to the host, then back through the board to the UART. The UARTS are not placed in “remote loopback” mode.

Except for unexpected errors due to low-level problems, this test always “passes”, because no special checking is performed. Like the Data Transmission Test, this test is provided to aid in signal tracing. You would connect a terminal to a selected channel and then hit keys on the terminal to generate data at will.

T49 Baud Rate Accuracy Test

Default passes:	1
Maximum passes:	1
Description:	Sends characters at 50 baud and times them check the baud rate accuracy.
Errors (fatal):	High-level preamble errors (see above) Failure to specify the interrupt (i option). Timeout sending a command
Errors, (ON):	Incorrect Baud Rate
Errors, (OE or OT):	Error: baud rate is too slow: X sent in y seconds. Error: baud rate is too fast: X sent in y seconds
Loadware:	FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non-PCI cards only)

Other associated arguments:

C	specified channels to test
O	option affects error messages

Restrictions: IntelliPort-IIEX needs boxes attached.

Summary:

After the standard loadware has been written and initialized, all selected UART channels are placed in normal mode. Then all the selected channels are initialized to 50 baud, and 60 characters are sent to each port. This rate is so slow, that the baud rate will be the absolute limiting factor regardless of how many ports are selected.

Then the host's real-time clock is used to time (in one-second granularity) how long it takes for this data to be all sent. If the time is more than a couple seconds off either way, an error is reported.

This test is not intended to catch micro-inaccuracies in crystal frequencies. The main point is to ensure there is no large systematic error resulting from installing the wrong oscillator, for example.

T50 Variable Rate (Wy60) Data Transmission Test

Default passes:	2 seconds
Maximum passes:	30000 seconds
Description:	Sends a string of data continuously to the UART at changing baud rates.
Errors (fatal):	High-level preamble errors (see above Failure to specify the interrupt (i option). Timeout sending a command
Loadware:	FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non-PCI cards only)

Other associated arguments:

C	specified channels to test
O	option affects error messages

Restrictions: IntelliPort-IIEX needs boxes attached.

Summary:

After the standard loadware has been written and initialized, all selected UART channels are placed in normal mode. Then all selected channels are initialized to 2400 baud, 1 stop bit, even parity, 7 bit characters, and xon/xoff protocol.

Test data is sent to the screen at this baud rate, then a special escape sequence, recognized by the WY60 as a command to change the rate to 9600 baud. Then the channels are changed to 9600 baud, and more test data is sent.

Similarly, we change to 19,200 baud and 38,400. Then we change back to 2400 again. If our specified time has elapsed, we end here, at the same baud rate we started. Otherwise, we continue spinning through the baud rates: 2400, 9600, 19200, and 38400.

This test is used to generate test data for FCC (and similar) emissions measurements.

T51 Variable Rate (Link MC60) Data Transmission Test

Default passes:	2 seconds
Maximum passes:	30000 seconds
Description:	Sends a string of data continuously to the UART at changing baud rates.
Errors (fatal):	High-level preamble errors (see above) Failure to specify the interrupt (i option). Timeout sending a command
Loadware:	FF.LOD (standard)

Required command line arguments:

I	to specify the interrupt (non-PCI cards only)

Other associated arguments:

C	specified channels to test
O	option affects error messages

Restrictions: IntelliPort-IIEX needs boxes attached.

Summary:

This test is identical to T50, except that it cycles through six, not four, baud rates. This is because the Link MC6 terminal also supports the rates of 76,800 and 115,200 baud.

So the cycle for this test is 2400, 9600, 19200, 38400, 76800, and 115200.

Appendix F: The Help message: a Quick reference.

```
 ------------- i2diag version 3.02.0 -------------

i2diag [options] [protocols] t{n} t{n}...
[options] are one or more of the following:
     P{n}    -- {n} is number of whole passes to do.
                default = 1
     X       -- continue P-option passes even if error occurs
     A{n}    -- {n} is the base address of board (normally in hex).
                default = 0x308
     I{n}    -- {n} is the interrupt to test:
3, 4, 5, 7, 10, 11, 12, or 15
                If several interrupts are listed,
                the first is used in normal operation;
                the remainder only by the 'interrupt test'.
                Ranges are allowed, e.g., I4-7 I10-15
     C{n}   --  {n} is the selected channel; 0 through 63
                Ranges are allowed, e.g., C0-7 C16-31
                Applies to all tests which operate on a per-channel
                basis.  If not supplied, tests will be performed on all.
     Y     --   Suppresses all operator prompts unless essential
     E{x}  --   {x} specifies the type of board you are testing:
                X  = expandable
                X6 = expandable using 16-bit data interface
                X8 = expandable using  8-bit data interface
                N  = non-expandable
                 (default: determined from power-on reset message)
     B{x}  --  {x} specifies the type of bus
                I = ISA (default)
                M = Micro Channel
                E = EISA
                P = PCI
     F     --   Stop test after the first error
     Z     --   Double all delays
     ,     --   Pause for operator after any error
     M     --   Mark parity/framing/overruns with _
     MM    --   Report parity/framing/overruns using status packets
     G{file} -- Log errors to specified file
     O{x...} -- Turn on/off output options
                {x...} is a list of letters; each one turns on an option:
         N -- Display Normal messages.
         E -- Display Detailed Loopback Errors.
         V -- Display Version numbers.
         P -- Display Protocols.
         T -- Display Testing Time.
         I -- Display IRQ Map.
         C -- Display Channel Map.
         S -- Display Test information at startup.
```

```
          F -- Display Detailed FIFO errors.


[protocols] may be given to restrict the tests to a specific
     baud rate, character size, parity, etc.  If not specified,
     protocols will be chosen randomly
Protocols and descriptions:
     -7          --  7 bits per character
     -8          --  8 bits per character
     -1          --  1   stop stop bit(s)
     -1.5        --  1.5 stop stop bit(s)
     -2          --  2   stop stop bit(s)
     -none       --     no parity
     -odd        --    odd parity
     -even       --   even parity
     -space      --  space parity
     -mark       --   mark parity
     -50         --       50  baud
     -75         --       75  baud
     -110        --      110  baud
     -134.5      --      134+ baud
     -150        --      150  baud
     -200        --      200  baud
     -300        --      300  baud
     -600        --      600  baud
     -1200       --     1200  baud
     -1800       --     1800  baud
     -2000       --     2000  baud
     -2400       --     2400  baud
     -3600       --     3600  baud
     -4800       --     4800  baud
     -7200       --     7200  baud
     -9600       --     9600  baud
     -19200      --   19,200  baud
     -38400      --   38,400  baud
     -56000      --   56,000  baud
     -57600      --   57,600  baud
     -64000      --   64,000  baud
     -76800      --   76,800  baud
     -115200     --  115,200  baud
     -153600     --  153,600  baud
     -230400     --  230,400  baud
     -307200     --  307,200  baud
     -460800     --  460,800  baud
     -921600     --  921,600  baud
     -o          --        no flow control
     -x          --  xon/xoff flow control

Tests are selected by number; e.g., t1 t2 t3
     Ranges are allowed, e.g., t1-10
     A number following a decimal point overrides the default
     number of passes for that test, e.g., t5.7 runs seven passes of test 5.
```

```
     Do not specify passes in the same argument as a range.
     Tests are performed in the order first listed.  If listed
     multiple times, they only performed the first time, but
     the last number of passes specified is used.
     For example, t4-8 t5.3 performs tests 4 through 8 in order
     performing only three passes of test 5.
     To eliminate selected tests from a range, select it afterward
     with zero passes.  For example t4-8 t5.0 performs tests 4, 6, and 8


Tests & Descriptions:
t1   Fifo Sanity test
       (default 1, maximum 1 pass)
     Host checks fifo register reliability
     Comments:
       Any error stops test.
       Non-expandable only.
       Requires EX or EN argument on command line
t2   Power-on Reset message test
       (default 1, maximum 1 pass)
     Host interprets power-on reset message
     Comments:
t4   Interactive LCD Controller test
       (default 1, maximum 1 pass)
     Operator validates correct LCD controller operation
     Comments:
       Any error stops test.
       Expandable only.
       Uses special loadware.
t6   Fifo Measurement test
       (default 1, maximum 1 pass)
     Host measures fifo size
     Comments:
       Any error stops test.
       Uses special loadware.
t7   External Bus Exerciser test
       (default 2, maximum 30000 seconds)
     Writes & reads devices on external bus
     Comments:
       Uses special loadware.
t8   Debug-Port Register test
       (default 20, maximum 30000 passes)
     Verifies access to debug port registers
     Comments:
       Uses special loadware.
       Needs debug port (power off before attaching.)
t10  Fifo Mailbox test
       (default 40, maximum 30000 passes)
     Host checks mailbox data from board & vice versa
     Comments:
       Any error stops test.
       Uses special loadware.
```

```
t11  Interrupt test
       (default 1000, maximum 15000 passes)
     Board generates selected interrupts to host
     Comments:
       Any error stops test.
       Uses special loadware.
t12  Random-pattern Board Memory test
       (default 5, maximum 30000 passes)
     Board performs random-pattern test on its local memory
     Comments:
       Any error stops test.
       Uses special loadware.
t13  FIFO Bi-Directional Loopback test
       (default 250, maximum 30000 passes)
     Host writes, reads data; board loops it back; both check integrity
     Comments:
       Any error stops test.
       Uses special loadware.
t14  Host Interface Timing Stress FIFO Loopback test
       (default 200, maximum 30000 passes)
     Host writes, reads, checks fifo data; (board loops it back)
     Comments:
       Any error stops test.
       Uses special loadware.
t15  FIFO Block-Message Loopback test
       (default 300, maximum 30000 passes)
     Host writes, reads, checks variable-length blocks of data
     Comments:
       Any error stops test.
       Uses special loadware.
t16  Board/Box Hardware Register test
       (default 5, maximum 30000 passes)
     Board checks Hardware Register reliability
     Comments:
       Any error stops test.
       Uses special loadware.
t18  FIFO Loopback Using DMA test
       (default 15, maximum 30000 passes)
     Board writes and reads FIFO using DMA, compares data
     Comments:
       Any error stops test.
       Expandable only.
       Uses special loadware.
t19  Walking-Bit Board Memory test
       (default 1, maximum 30000 pass)
     Board performs walking-bit test on its local memory
     Comments:
       Any error stops test.
       Uses special loadware.
t23  AMCC operation register checkout test
       (default 5, maximum 30000 passes)
     Host ensures it can write and read AMCC interface registers
```

```
     Comments:
       Any error stops test.
t24  POS register checkout test
       (default 5, maximum 30000 passes)
     Host ensures it can write and read Micro Channel POS registers
     Comments:
       Any error stops test.
t25  Internal Data Loopback test
       (default 3, maximum 30000 passes)
     Sends data to uart, receives, compares
     Comments:
       Channels may be selected on command line.
t26  External Data Loopback (DSS up) test
       (default 10, maximum 30000 passes)
     Sends data to uart, through loopback plug, receives, compares
     Comments:
       Channels may be selected on command line.
       Requires loopback plug:
          TxD(3)->RxD(2) | DTR(8)->DCD(20) | RTS(5)->CTS(4)
t27  External Data Loopback (DSS down) test
       (default 10, maximum 30000 passes)
     Sends data to uart, through loopback plug, receives, compares
     Comments:
       Channels may be selected on command line.
       Requires loopback plug:
          TxD(3)->RxD(2) | DTR(8)->DCD(20) | RTS(5)->CTS(4)
t28  External Data Loopback w/Data Set Exercise test
       (default 10, maximum 30000 passes)
     Sends data to uart, through loopback, receives, compares, moves DTR &
RTS
     Comments:
       Channels may be selected on command line.
       Requires loopback plug:
          TxD(3)->RxD(2) | DTR(8)->DCD(20) | RTS(5)->CTS(4)
t30  External Data Loopback (RTS/CTS Flow Ctrl) test
       (default 20, maximum 30000 passes)
     Sends data to uart, through loopback plug, receives, compares
     Comments:
       Channels may be selected on command line.
       Requires loopback plug:
          TxD(3)->RxD(2) | DTR(8)->DCD(20) | RTS(5)->CTS(4)
t31  Data Leak Test test
       (default 1, maximum 1 pass)
     Sends some data to UART, ensures that _none_ returns
     Comments:
       Channels may be selected on command line.
       Requires loopback plug:
          TxD(3)->RxD(2) | DTR(8)->DCD(20) | RTS(5)->CTS(4)
t34  External Data Loopback (DTR/CTS Flow Ctrl) test
       (default 20, maximum 30000 passes)
     Sends data to uart, through loopback plug, receives, compares
     Comments:
```

```
       Channels may be selected on command line.
       Requires loopback plug:
          TxD(3)->Rxd(2) | DTR(8)->CTS(4) | RTS(5)->DCD(20)
t35  Break Generation/Detection test
       (default 10, maximum 30000 passes)
     Sends breaks & tries to receive them
     Comments:
       Channels may be selected on command line.
       Requires loopback plug:
          TxD(3)->RxD(2) | DTR(8)->DCD(20) | RTS(5)->CTS(4)
t36  DTR/DCD Loopback test
       (default 10, maximum 30000 passes)
     Ensures DCD tracks DTR changes on each channel
     Comments:
       Channels may be selected on command line.
       Requires loopback plug:
          TxD(3)->RxD(2) | DTR(8)->DCD(20) | RTS(5)->CTS(4)
t37  RTS/CTS Loopback test
       (default 10, maximum 30000 passes)
     Ensures CTS tracks RTS changes on each channel
     Comments:
       Channels may be selected on command line.
       Requires loopback plug:
          TxD(3)->RxD(2) | DTR(8)->DCD(20) | RTS(5)->CTS(4)
t40  DTR/DSR Loopback test
       (default 10, maximum 30000 passes)
     Ensures DSR tracks DTR changes on each channel
     Comments:
       Expandable only.
       Channels may be selected on command line.
       Requires loopback plug:
          DTR(8)->DSR(11) | RTS(5)->RI(22)
t41  RTS/RI  Loopback test
       (default 10, maximum 30000 passes)
     Ensures RI  tracks RTS changes on each channel
     Comments:
       Expandable only.
       Channels may be selected on command line.
       Requires loopback plug:
          DTR(8)->DSR(11) | RTS(5)->RI(22)
t43  DTR/CTS Loopback test
       (default 10, maximum 30000 passes)
     Ensures CTS tracks DTR changes on each channel
     Comments:
       Channels may be selected on command line.
       Requires loopback plug:
          TxD(3)->Rxd(2) | DTR(8)->CTS(4) | RTS(5)->DCD(20)
t44  RTS/DCD Loopback test
       (default 10, maximum 30000 passes)
     Ensures DCD tracks RTS changes on each channel
     Comments:
       Channels may be selected on command line.
```

```
       Requires loopback plug:
          TxD(3)->Rxd(2) | DTR(8)->CTS(4) | RTS(5)->DCD(20)
t45  Data Transmission test
       (default 2, maximum 30000 seconds)
     Sends a string over and over to each selected channel
     Comments:
       Channels may be selected on command line.
t46  DTR Tracing test
       (default 2, maximum 30000 seconds)
     Raises/Lowers DTR repeatedly on selected channels
     Comments:
       Channels may be selected on command line.
t47  RTS Tracing test
       (default 2, maximum 30000 seconds)
     Raises/Lowers RTS repeatedly on selected channels
     Comments:
       Channels may be selected on command line.
t48  Data Echo test
       (default 2, maximum 30000 seconds)
     Reads and echos incoming characters
     Comments:
       Channels may be selected on command line.
t49  Baud Rate Accuracy test
       (default 1, maximum 1 pass)
     Sends 60 characters at 50 baud and times them
     Comments:
       Channels may be selected on command line.
t50  Variable Rate (Wy60) Data Transmission test
       (default 2, maximum 30000 seconds)
     Sends messages at 2400, 9600, 19200, 38400 baud
     Comments:
       Channels may be selected on command line.
t51  Variable Rate (Link MC6) Data Transmission test
       (default 2, maximum 30000 seconds)
     Sends messages at 2400, 9600, 19200, 38400, 76800, 115200 baud
     Comments:
       Channels may be selected on command line.


Numbers used in options or to identify tests may be
given in octal, decimal, or hexadecimal notation.

Numbers beginning with ‘0x’ are taken as hexadecimal.
This is the natural way to express board addresses, e.g. 0x308.
Numbers beginning with ‘0’ are taken as octal.
Numbers beginning with other digits are taken as decimal.
```