Traditional boundary scan tools are very effective in manufacturing test for diagnosing board problems - especially when the board is designed to optimize the use of these tools. In concept it is very simple - you hand the tools your net list, they automatically calculate all the necessary test vectors, apply the vectors, read/interpret the results and tell you where the discrepancies are on your board.

In practice, though, setting this up can be a very tedious and time consuming exercise - especially if the board is a prototype that has no test fixtures, an unstable net list and questionable CAD data after numerous undocumented cuts and jumpers.

When debugging prototypes you are usually trying to figure out what is going on in a very small area of the board and don't really want to take the time or trouble to setup all the test vectors, etc. - you just want to see if a few pins have continuity and you want the answer right now.

A new genre of boundary scan tools is available now that remove the tedium of traditional boundary scan test. Within seconds, you can manually see and control every scan enabled pin under every JTAG device on your board in real time without having to create test vectors, test executives, test projects, etc. You just drop the parts on the screen, connect the chain to your PC and hit SCAN.

JTAG Review from a New Perspective

The idea behind boundary scan is simple enough – connect a latch to every I/O and buffer enable on a device so you can control and monitor the I/Os independently from the internal logic. Connect those latches in a long shift register so you can shift the results in/out. The result looks like Figure 1.

![Figure 1](image.png)

The yellow boxes represent the latching logic – scan cells - that are used to capture and monitor the I/O in and out of the device.

Usually, these are transparent and the user and the device don't even know they are there.

In this mode, called SAMPLE/PRELOAD, you instruct the yellow boxes to unobtrusively take a snapshot of the signals and then shift the results out TDO to view them.

This sampling and shifting operation is completely independent of the operation of the device and internal logic. You can do this all day long.

Figure 1. Yellow boxes contain logic to monitor and control I/Os independently from internal logic.
without affecting the device’s operation in any way. If you do this over and over and display the
result in the same form factor as your device, then you can see a real-time display of pin activity.

The beauty of this is you can now instantly see if a signal is high, low or toggling – even if the
signal is buried under a BGA package and while your circuit is running. For example, here is a
JTAG chain of a configuration PROM and an FPGA:

In this screen shot you see the parallel port is connected via JTAG to the configuration PROM,
which is connected to the FPGA which is connected back to the parallel port. The red, black and
blue dots represent the pins on the parts. Red indicates a pin is at a logic high, black indicates it
is at a logic low and blue indicates pins you can’t scan with boundary scan (power, ground, the
JTAG signals themselves, etc).

If a pin is toggling between red and black, then it is actively being driven. All of the sudden it has
become quick and easy to see the status of a signal. You just look at the pin – if it is blinking then
there is activity on that pin, if it is red the signal is high and if it is black the signal is low.

Can you measure a 12MHz oscillator with boundary scan? No, boundary scan isn’t fast enough
for that. Remember, you have to tell the scan cells to capture the values, then shift them all out,
then capture again, then shift out again, etc. This takes way too long to do any high speed
measurements. But, let’s face it, most of the time when you connect an O-Scope to a signal you
just want to know if the signal is high, low or toggling – you are looking for an indication of activity
– and that is exactly what these tools provide, except they can do it in places where the O-Scope can’t get to!

**Controlling pins**

The best news is boundary scan isn’t just passive. In addition to the SAMPLE/PRELOAD mode we discussed above, there is also an EXTEST mode where you tell the scan cells (the yellow boxes in Figure 1.) to disconnect the signals from the internal logic and allow the scan cells to control the I/Os directly.

In EXTEST you shift data into the scan cells via TDI, then tell the scan cells to apply the data to the I/Os. You now have total control over every JTAG enabled I/O on the device. You can enable buffers, drive buffers, and monitor signals simply by putting 1’s and 0’s into the appropriate scan cells.

How do you control the values that get put into the scan cells? Simple – connect a virtual switch. We’ll use this concept in the next several sections to show you a new way to test and debug your boards.

**Using Boundary Scan for Low Level Debug**

*Simple continuity testing*

The example in Figure 3 shows a Motorola processor and an ASIC in the JTAG chain.

![Figure 3: A Motorola processor and an ASIC. Virtual Switches and LEDs are used to monitor and control the address bus pins.](image-url)
In this example we have connected virtual 7-segment LEDs to the Data and Address buses so we can see what they are doing at a glance. On the right side there are 4 columns of devices: the first column are toggle switches and are connected to the output buffer enables on the address bus of the processor. They are lit, indicating a logic 1 is being applied to the enable. The second column of devices are momentary switches that are connected to the input of the output buffer. The third column of devices are virtual LEDs connected to the input buffers of the processor, and the fourth column of devices are virtual LEDs connected to the input buffers on the ASIC.

To do a simple continuity test, we just set the buffer enables, drive the output buffers with the momentary switches and watch the LEDs. If both LEDs light up, then the input buffers on both devices saw the logic level change and we know we have continuity.

In figure 3 you can see where we pressed the momentary switch on the 2\textsuperscript{nd} row to drive an address line and both LEDs lit up. We have continuity. When we pressed the switch on the 4\textsuperscript{th} row, only the Processor LED lit up – the signal didn’t make it to the ASIC.

That’s all there is to it – just click on switches to put values in the scan cells, and watch the results on LEDs. Quick and easy.

Another example: The yellow LED on the ASIC is connected to the 80MHz Oscillator. We see the Virtual LED toggling so we know the oscillator is connected to the ASIC. If we saw it stuck at a high or a low, then we would know we have a problem.

\textit{I/O Testing}

One of the problems with most boards is testing IO’s without a test fixture. Most of the time you just want to see if a pin under a BGA is physically connected to an I/O on your board. With these tools it is easy – just toggle the pin manually and monitor the pin on the connector with an O- Scope or DVM.

\textit{Control Panel}

Once you have mapped out your test circuit, you can rearrange the virtual switches and LEDs on the screen to create a quick and dirty control panel from your board. You can even use this to drive/control circuitry before firmware is written. Activate a solenoid, turn on a light, drive a D/A, etc.

\textit{Flash Programming}

Since boundary scan gives you total control over the pins on a device, and if you have a Flash device connected to those pins, why not use boundary scan to program the Flash device? This is a great way to get access to a Flash memory without having to provide extra circuitry or means to remove it from the circuit. You just tell the tool which pins in the chain are connected to the Flash Device, specify a data file and hit PROGRAM. The tool automatically configures the device, erases it, programs it and verifies it.

This is a great way to independently program your Flash memory in system without a functioning processor to do it. This is a great way to get a boot loader into a Flash memory for a processor, then let the processor come up and load the rest of the memory.

Which brings up the downside of Flash programming with Boundary Scan – it can be VERY slow. A 10K-Byte file can take a few seconds to several minutes (depending on the tools, options and length of your scan chain).
Here’s the problem – we have to shift the entire scan chain just to change the state of a single pin. That means to write a single byte we have to:

- Shift the entire chain to setup address and data
- Shift the entire chain to enable the write-enable
- Shift the entire chain to disable the write-enable.

If your typical scan chain is a couple thousand bits (usually 3 scan cells per pin, times the number of pins) then you will have to do 5 to 10 thousand shifts just to write one byte. Repeat that for every byte in your file and you can see why it will take some time.

The good news is new faster USB based pods are coming out that will drastically reduce this programming time. The traditional boundary scan tools also offer several high speed programming options and plug-in modules.

**SPI Flash Programming**

Same story as above – simply tell the tools how the SPI device is connected to the JTAG chain, specify a data file and hit PROGRAM. Quick, easy, simple.

**Scripting**

Punching all those little virtual switches all the time gets real tedious real fast. Fortunately, you can record a test session on a known good board, then swap boards and hit PLAY to repeat the exact same test. Using this you can quickly build a little prototype test fixture for your early boards without having to worry about test-vectors, CAD data, ...
You Need Both

It is important to understand that these tools do not replace the traditional boundary scan tools - they augment them. These low level manual debug tools are great for prototype debug in the lab and for learning about boundary scan (and convincing management to invest in the high end traditional tools!) because they are quick, simple, easy and inexpensive. The high-end automated tools are great for diagnosing high volume mature boards in production – especially if those boards are properly designed for automated test!

The key difference is that these new low level tools provide real-time display and control of a board while it is running. The traditional tools typically do full board scans on boards that are not running.

For More Information …

The examples used in this article are from Ricreations “Universal Scan” (www.UniversalScan.com) and Macraigor Systems “J-Scan” (www.j-scan.com). Check out those websites for more examples, free trials and tutorials.