



## Boundary Scan in XC4000 and XC5000 Series Devices

XAPP 017 July 15, 1996 (Version 1.1)

Application Note

### Summary

XC4000 and XC5000 Series FPGA devices contain boundary-scan facilities that are compatible with IEEE Standard 1149.1. This Application Note describes those facilities in detail, and explains how boundary scan is incorporated into an FPGA design.

### Xilinx Family

XC4000 Series, XC5200

### Demonstrates

Boundary Scan

## Introduction

In production, boards must be tested to assure the integrity of the components and the interconnections. However, as integrated circuits have become more complex and multi-layer PC-boards have become more dense, it has become increasingly difficult to test assembled boards.

Originally, manufacturers used functional tests, applying input stimuli to the input connectors of the board, and observing the results at the output. Later, "bed-of-nails" testing became popular, where a customized fixture presses sharp, nail-like stimulus- and test-probes into the exposed traces on the board. These probes were used to force signals onto the traces and observe the response.

However, increasingly dense multi-layer PC boards with ICs surface-mounted on both sides have stretched the capability of bed-of-nail testing to its limit, and the industry is forced to look for a better solution. Boundary-scan techniques provide that solution.

The inclusion of boundary-scan registers in ICs greatly improves the testability of boards. Boundary scan provides a mechanism for testing component I/Os and interconnections, while requiring as few as four additional pins and a minimum of additional logic in each IC. Component testing may also be supported in ICs with self-test capability.

Devices containing boundary scan have the capability of driving or observing the logic levels on I/O pins. To test the external interconnect, devices drive values onto their outputs and observe input values received from other devices. A central test controller compares the received data with expected results. Data to be driven onto outputs is distributed through a chain of shift registers, and observed input data is returned through the same shift-register path.

Data is passed serially from one device to the next, thus forming a boundary-scan path or loop that originates at the test controller and returns there. Any device can be temporarily removed from the boundary-scan path by bypassing its internal shift registers, and passing the serial data directly to the next device.

XC4000/XC5000 FPGA devices contain boundary-scan registers that are compatible with the IEEE Standard 1149.1, that was derived from a proposal by the Joint Test Action Group (JTAG). External (I/O and interconnect) testing is supported; there is also limited support for internal self-test.

## Overview of XC4000/XC5000 Boundary-Scan Features

XC4000/XC5000 devices support all the mandatory boundary-scan instructions specified in the IEEE Standard 1149.1. A Test Access Port (TAP) and registers are provided that implement the EXTEST, SAMPLE/PRELOAD and BYPASS instructions. The TAP can also support two USERCODE instructions.

**Note:** If boundary scan is not used after the device is configured, the user can use the special boundary scan pads as input or output pins. And like the regular IOBs, these input and output pins have pullups and pulldowns available. The TDI, TMS, and TCK pads can be used as input pads. The TDO pad can be used as an output pad.

Boundary-scan operation is independent of individual IOB configuration and package type. All IOBs are treated as independently controlled bidirectional pins, including any unbonded IOBs. Retaining the bidirectional test capability even after configuration affords tremendous flexibility for interconnect testing.

Additionally, internal signals can be captured during EXTEST by connecting them to unbonded IOBs, or to the unused outputs in IOBs used as unidirectional input pins. This partially compensates for the lack of INTEST support.

The public boundary-scan instructions are always available prior to configuration. After configuration, the public instructions and any USERCODE instructions are only available if specified in the design. While SAMPLE and BYPASS are available during configuration, it is recommended that

boundary-scan operations not be performed during this transitory period.

In addition to the test instructions outlined above, the boundary-scan circuitry can also be used to configure the FPGA device, and readback the configuration data.

The following description assumes that the reader is familiar with boundary-scan testing and the IEEE Standard. Only issues specific to the XC4000/XC5000 implementation are discussed in detail. For general information on boundary scan, please refer to the bibliography.

## Deviations from the IEEE Standard

The XC4000/XC5000 boundary scan implementation deviates from the IEEE standard in that three dedicated pins (CCLK, PROGRAM and DONE) are not scanned.

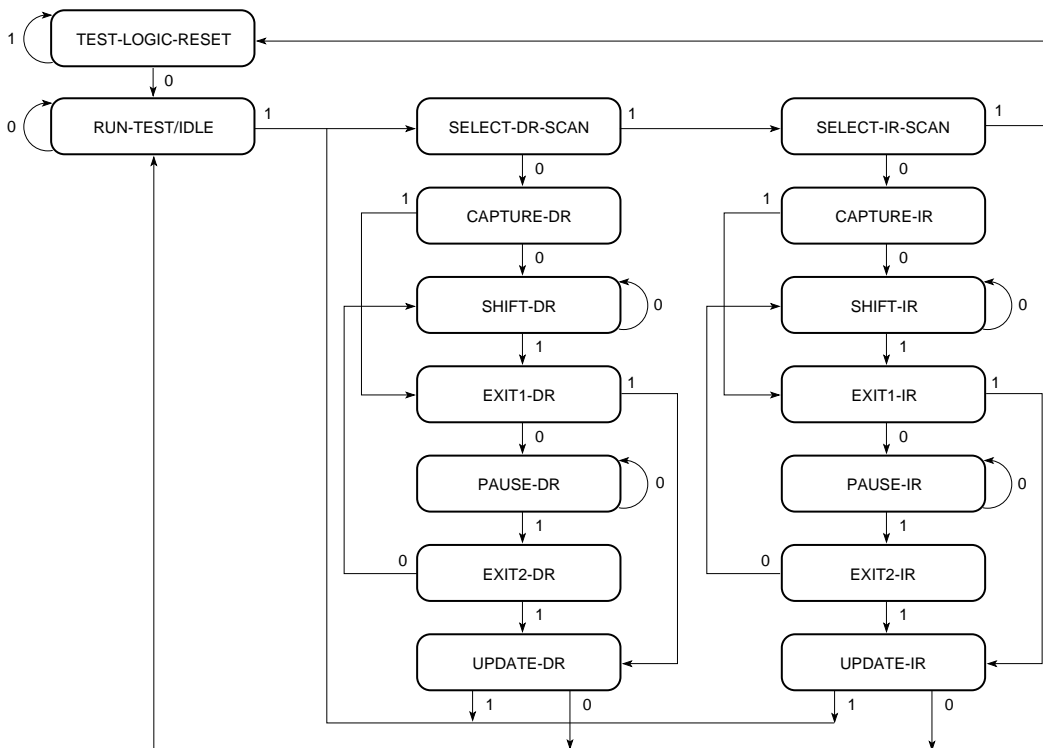
It should also be noted that the Test Data Register contains three Xilinx test bits (BSCANT.UPD, TDO.O and TDO.T)

and that bits of the register may correspond to unbonded or unused pins.

Additionally, the EXTEST instruction incorporates INTEST-like functionality that is not specified in the standard, and system clock inputs are not disabled during EXTEST, as recommended in the standard.

The TAP pins (TMS, TCK, TDI and TDO) are scanned, but connections to the TAP controller are made before the boundary-scan logic. Consequently, the operation of the TAP controller cannot be affected by boundary-scan test data.

When the TAP is in the shift-DR state the contents of all data registers are shifted; if you are in the middle of shifting out data from the data register in a XC4000, complete shifting out of all data first, before switching to the instruction or bypass register.



NOTE: The value shown adjacent to each state transition in this figure represents the signal present at TMS at the time of a rising edge at TCK.

X2680

Figure 1: State Diagram for the TAP Controller

## Boundary-Scan Hardware Description

### Test Access Port

The boundary-scan logic is accessed through the Test Access Port (TAP), which comprises four semi-dedicated pins: Test Mode Select (TMS), Test Clock (TCK), Test Data Input (TDI) and Test Data Output (TDO), as defined in the IEEE specification.

The TAP pins are permanently connected to the boundary-scan circuitry. However, once the device is configured, the connections may be ignored unless the use of boundary scan is specified in the design (See "Using Boundary Scan").

If the use of boundary scan is specified, the TAP input pins (TMS, TCK and TDI) may still be shared with other logic, subject to limitations imposed by external connections and the operation of the TAP Controller. In designs that do not use boundary scan after configuration, the TAP pins can be used as inputs or outputs from the user logic in the FPGA device. TMS, TCK and TDI are available as unrestricted I/Os, while TDO only provides a 3-state output.

### TAP Controller

The TAP Controller is a 16-state state machine that controls the operation of the boundary-scan circuitry in response to TMS. This state machine implements the state diagram specified by the IEEE standard (Figure 1) and is clocked by TCK.

Upon power-on, or if the boundary scan logic is not used in the application, the TAP controller is forced into the Test-Logic-Reset state. After configuration, the controller remains disabled, unless its use is explicitly specified in the user design. PROGRAM resets the latched decodes for EXTEST, CONFIGURE, and READBACK instructions.

Loading a 3-bit instruction into the Instruction Register (IR) determines the subsequent operation of the boundary-scan logic, Table 1. The instruction selects the source of the TDO pin, and selects the source of device input and output data (boundary-scan register or input pin/user logic)

Note: In the XC4000, whenever the TAP Controller is in the Shift-DR state, all data registers are shifted, regardless of the instruction. DR data is modified even if a BYPASS instruction is executed.

The instruction register is not used only to hold the current instruction. If the TAP is in the capture-IR state and TCK goes high, the instruction register captures the current boundary-scan state of the device.  $I_0$  is 1 by default.  $I_1$  is 0 by default.  $I_2$  is 0 if the device is in configure by boundary scan mode. Before and after configure by boundary scan mode,  $I_2$  will capture 1. Note that  $I_0$  is shifted out of TDO first, then  $I_1$ , and then  $I_2$ .

Table 1: Boundary Scan Instructions.

Instruction	$I_2$	$I_1$	$I_0$	Test Selected	TDO Source	I/O Data Source
0	0	0	0	EXTEST	DR	DR
0	0	1		SAMPLE/ PRELOAD	DR	Pin/Logic
0	1	0		USER 1	TDO1	Pin/Logic
0	1	1		USER 2	TDO2	Pin/Logic
1	0	0		READBACK	Readback Data	Pin/Logic
1	0	1		CONFIGURE	DOUT	Disabled
1	1	0		RESERVED	—	—
1	1	1		BYPASS	Bypass Reg	Pin/Logic

$I_0$  is closest to DTO

### The Boundary-Scan Data Register

The Data Register (DR) is a serial shift register implemented in the IOBs of the FPGA device, (Figure 2). Potentially, each IOB can be configured as an independently controlled bidirectional pin. Therefore, three data register bits are provided per IOB: for input data, output data and 3-state control. In practice, many of these bits are redundant, but they are not removed from the scan chain.

An update latch accompanies each bit of the DR, and is used to hold injected test data stable during shifting. The update latch is opened during the Update-DR state of the TAP Controller when TCK is Low.

In a typical DR instruction, the DR captures data during the Capture-DR state (on the rising edge of TCK). This data is then shifted out and replaced with new test data. Subsequently, the update latch opens, and the new test data becomes available for injection into the logic or the interconnect. The injection of data occurs only if an EXTEST instruction is in progress.

Note: The update latch is opened whenever the TAP Controller is in the Update-DR state, regardless of the instruction. Care must be exercised to ensure that appropriate data is contained in the update latch prior to initiating an EXTEST. Any DR instruction, including BYPASS, that is executed after the test data is loaded, but before the EXTEST commences, changes the test data.

The IEEE Standard does not require the ability to inject data into the on-chip system logic and observe the results during EXTEST. However, this capability helps compensate for the lack of INTEST. Logic inputs may be set to specific levels by a SAMPLE/PRELOAD or EXTEST instruction and the resulting logic outputs captured during a subsequent EXTEST. It must be recognized, however, that all DR bits are captured during an EXTEST and, therefore, may change.

Pull-up and pull-down resistors remain active during boundary scan. Before and during configuration, all pins

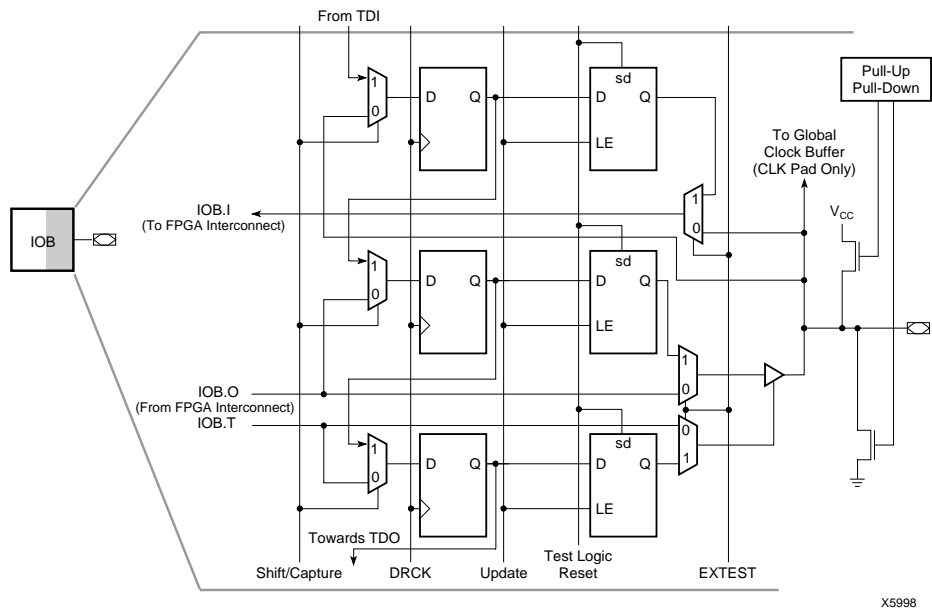


Figure 2: Boundary Scan Logic in a Typical IOB

are pulled up. After configuration, the IOB can be configured with a pull-up resistor, a pull-down resistor or neither.

Note: Internal pull-up/pull-down resistors must be taken into account when designing test vectors to detect open circuit PC traces.

The primary and secondary global clock inputs (PGCK1-4 and SGCK1-4) are taken directly from the pins, and cannot be overwritten with boundary-scan data. However, if necessary, it is possible to drive the clock input from boundary scan. The external clock source is 3-stated, and the clock net is driven with boundary scan data through the output driver in the clock-pad IOB. If the clock-pad IOBs are used for non-clock signals, the data may be overwritten normally.

Figure 3 shows the data-register cell for a TAP pin. An OR-gate permanently disables the output buffer if boundary-scan operation is selected. Consequently, it is impossible for the outputs in IOBs used by TAP inputs to conflict with TAP operation. TAP data is taken directly from the pin, and cannot be overwritten by injected boundary-scan data.

Table 2 lists, in data-stream order, the boundary-scan cells that make up the DR. The cell closest to TDO corresponds to the first bit of the data-stream, and is at the top of the table. This order is consistent with the BSDL description.

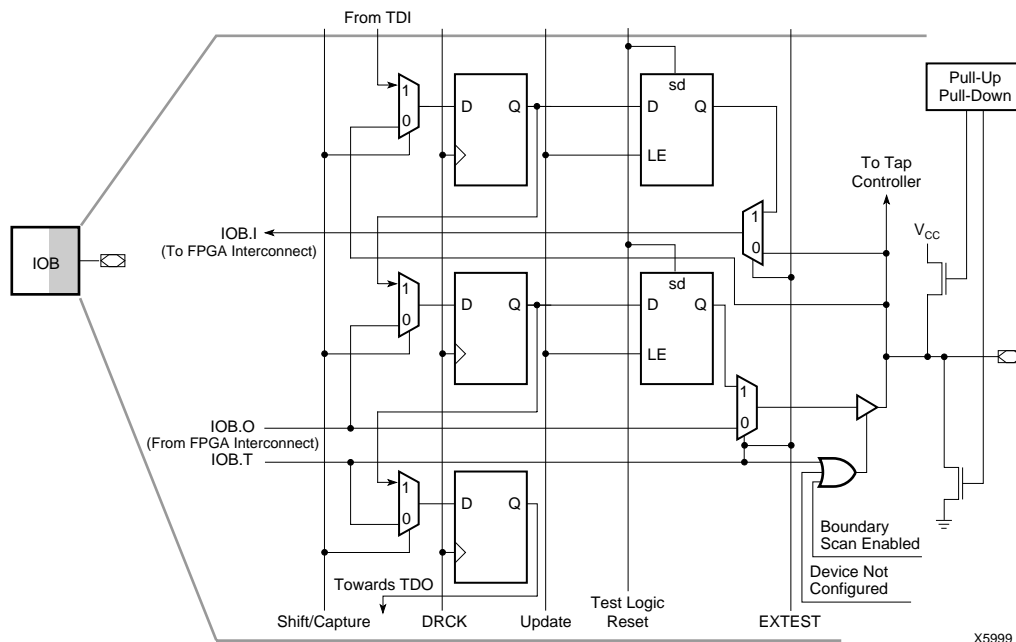
Each IOB corresponds to three bits in the DR. The 3-state control is first (closest to TDO), the output is next, and the input is last. Other signals correspond to individual register bits. IOB locations assume that the die is viewed from the top, as in XDE.

Note: All IOBs remain in the DR, independent of whether they are actually used, or even bonded. Three bits, BSCANT.UPD, TDO.O and TDO.T, are included for Xilinx test purposes, and may be ignored by other users. CCLK, PROGRAM and DONE are not included in the boundary scan.

Table 2: Boundary Scan Order

Bit 0 ( TDO end)	TDO.T
Bit 1	TDO.O
Bit 2	{ Top-edge IOBs (Right to Left)
	{ Left-edge IOBs (Top to Bottom)
	MD1.T
	MD1.O
	MD1.I
	MD0.I
	MD2.I
	{ Bottom-edge IOBs (Left to Right)
	{ Right-edge IOBs (Bottom to Top)
	B SCANT.UPD
( TDI end)	

X2674



**Figure 3: Boundary Scan Logic in a TAP Input (TMS, TCK, and TDI Only)**

Tables in the data sheet show the DR order for all XC4000 family devices. The DR also includes the following non-pin bits: TDO.T and TDO.I, which are always bits 0 and 1 of the DR, respectively, and BSCANT.UPD which is always the last bit of the DR.

## The Bypass Register

This is a 1-bit shift register that passes the serial data directly to TDO when a bypass instruction is executed.

## User Registers

The XC4000 boundary-scan instruction set includes two USERCODE instructions, USER1 and USER2. Connections are provided to the TAP and TAP controller that, together with direct connections to the TAP pins, permit the user to include boundary-scan self-test features in the design.

The boundary scan block has six connections for user registers: SEL1, SEL2, TDO1, TDO2, DRCK and IDLE. TDI is available directly from the IOB that provides the TDI pin.

**Note:** The TDI signal supplied to user test logic is overwritten by boundary-scan test data during EXTEST. During user tests, it is not altered.

**SEL1, SEL2** – SEL1 and SEL2 enable user logic. They are asserted (High) when the instruction register contains instructions USER1 and USER2, respectively.

**TDO1, TDO2** – TDO1 and TDO2 are inputs to the TDO output multiplexer, permitting user access to the serial boundary-scan output. They are selected when executing the instructions USER1 and USER2, respectively. Input to user data registers can be derived directly from the TDI pin, thus completing the boundary-scan chain.

There is a one flip-flop delay between TDO1/TDO2 and the TDO output. This flip-flop is clocked on the falling edge of TCK.

**DRCK** – Data register clock (DRCK) is a gated and inverted version of TCK. It is provided to clock user test-data registers. TDI data should be sampled with the falling edge of DRCK (rising edge of TCK). The TDO output flip-flop accepts data on the rising edge of DRCK (falling edge of TCK). DRCK is active only during the Capture-DR and Shift-DR states of the TAP controller.

**IDLE** – IDLE is a second gated and inverted version of TCK. It is active during the Run-Test/Idle state of the TAP controller, and may be used to clock user test logic a set number of times, determined through TMS by the central test controller.

## Using Boundary Scan

Full access to the built-in boundary-scan logic is always available between power-up and the start of configuration. Optionally, the built-in logic is fully available after configuration if boundary scan is specified in the design. At this time, user test logic is also available, and may be accessed through the boundary-scan port. During configuration, a reduced boundary-scan capability remains available: the SAMPLE/PRELOAD and BYPASS instructions only.

Figure 4 is a flow chart of the FPGA start-up sequence that shows when the boundary-scan instructions are available. Since  $\overline{\text{PROGRAM}}$  resets the TAP controller, boundary-scan operations cannot commence until  $\overline{\text{PROGRAM}}$  has been taken High.

Full boundary-scan capabilities are then available until  $\overline{\text{INIT}}$  is High. Without external intervention,  $\overline{\text{INIT}}$  automatically goes High after  $\sim 1$  ms. If more time is required for boundary-scan testing,  $\overline{\text{INIT}}$  may be held Low beyond this period by applying an external Low signal to the  $\overline{\text{INIT}}$  pin until testing is complete.

Boundary scan can be accessed before the FPGA is configured and after the FPGA is configured. If you want to access boundary scan before the device is configured, then when you power-up the device, hold the  $\overline{\text{INIT}}$  pin low until VCC has risen to  $V_{cc}(\text{min})$ . If you have already started configuring the device, data frames are already being sent to the FPGA, then you have two choices. You can either access full-boundary scan mode, or limited boundary scan mode. If you want to access full-boundary scan mode, then both  $\overline{\text{INIT}}$  and  $\overline{\text{PROGRAM}}$  must be brought low (Hold  $\overline{\text{INIT}}$  and  $\overline{\text{PROG}}$  low for over 300 ns and then release  $\overline{\text{PROGRAM}}$ . After releasing  $\overline{\text{PROGRAM}}$ , continue to hold  $\overline{\text{INIT}}$  low while sending signals to the TAP. If you can use the limited boundary scan mode (which means you only can use the SAMPLE/PRELOAD and BYPASS instructions), then just bring  $\overline{\text{INIT}}$  low.

Accessing boundary scan after the device is configured has one requirement. The BSCAN symbol must be instantiated/inserted into your design with the correct syntax (see Figure 5). In this case, activating boundary scan after configuration amounts to toggling the TAP pins.

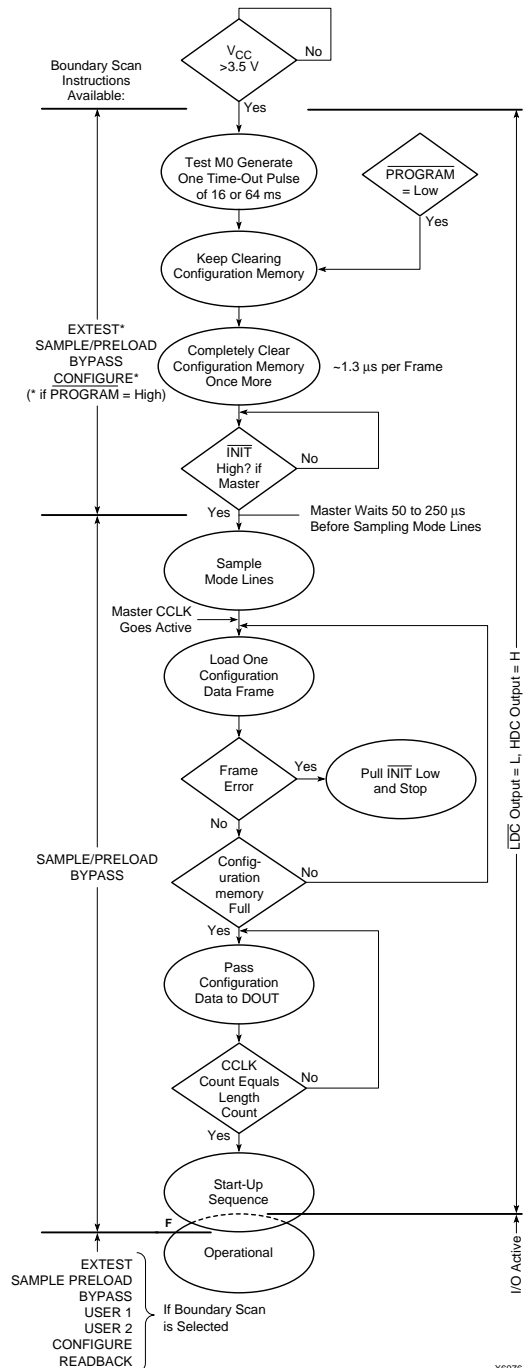
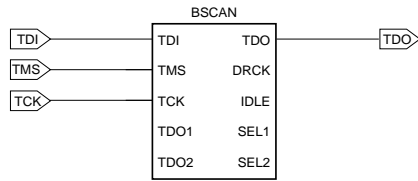
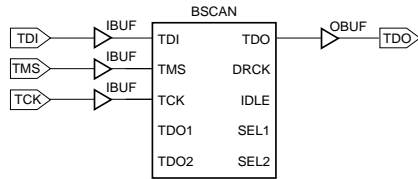


Figure 4: Start-up Sequence



**Figure 6: Typical Non-Boundary-Scan TDO Connection**

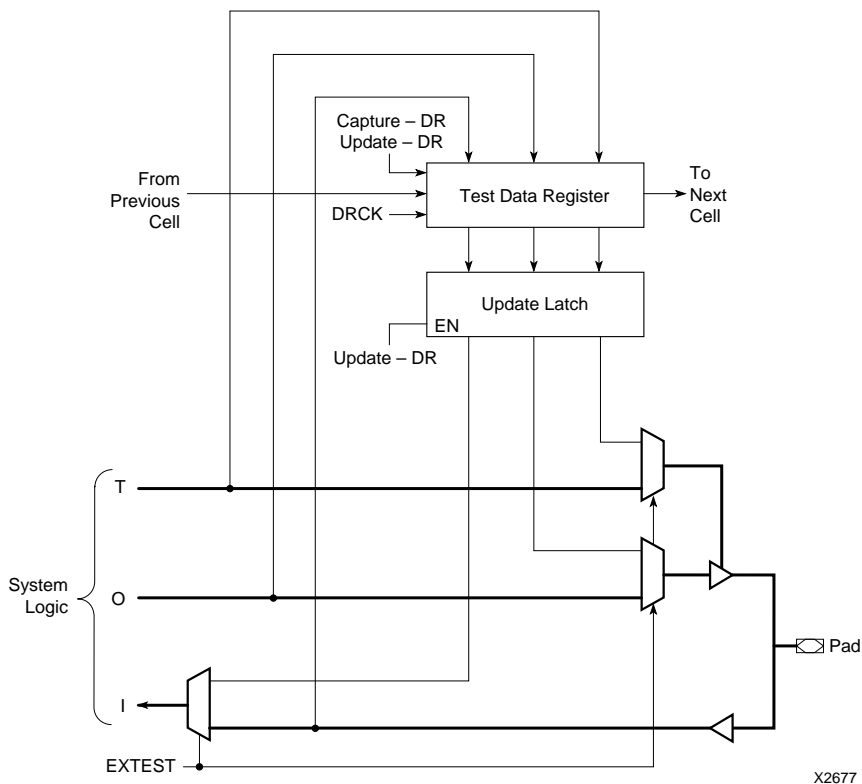
4k BSCAN Syntax for BSCAN after configure symbol



5k BSCAN Syntax for BSCAN after configure symbol

X5966

**Figure 5: Boundary-Scan Schematic**



X2677

**Figure 7: EXTEST Data Flow**

The IEEE definition of EXTEST only requires that test data be driven onto outputs, that 3-state output controls be overridden, and that input data be captured. The capture of output data and 3-state controls and the forcing of test data into the system logic is normally performed during INTEST.

The XC4000 effectively performs EXTEST and INTEST simultaneously. This added functionality permits the testing of internal logic, and compensates for the absence of a separate INTEST instruction. However, when performing an EXTEST, care must be taken over what signals are driven into the system logic; data captured from internal system logic must be masked out of the test-data stream before performing check-sum analysis.

**SAMPLE/PRELOAD** – The SAMPLE/PRELOAD instruction permits visibility into system operation by capturing the state of the I/O. It also permits valid data to be loaded into the update register before commencing an EXTEST.

The DR and update latch operate exactly as in EXTEST (see above). However, data flows through the I/O unmodified.

**BYPASS** – The BYPASS instruction permits data to be passed synchronously to the next device in the boundary-scan path. There is a 1-bit shift register between the TDI and TDO flip-flop.

**USER1, USER2** – These instructions permit test logic, designed by the user and implemented in CLBs, to be accessed through the TAP. Test clocks and paths to TDO are provided, together with two signals that indicate that user instructions have been loaded. For details, see the User Registers section above.

User tests depend upon CLBs and interconnect that must be configured to operate. Consequently, they may only be performed after configuration.

**CONFIGURE** – Steps to Follow to configure a Xilinx XC4000, XC4000E, or XC5200 via JTAG:

1. Turn 'on' the boundary scan circuitry.

This can be done one of two ways, either via powerup or via a configured device with boundary scan enabled. If you want to do this via powerup, then just hold the  $\overline{\text{INIT}}$  pin low when power is turned on. When  $V_{CC}$  has reached  $V_{CC}(\text{min})$ , then the TAP can be toggled to enter JTAG instructions. If you want to do this from a configured device, then just start toggling the JTAG port pins to go from test-logic-reset to run-test-idle.

2. Load the Xilinx Configure instruction into the IR.

The Xilinx Configure instruction is  $101(I_2 I_1 I_0)$ .  $I_0$  is the bit shifted in first into the IR.

3. After shifting in the Xilinx Configure instruction, make the Configure instruction the current JTAG instruction by going to the update-IR state. When TCK goes low in the

update-IR state, the FPGA is now in the JTAG configuration mode and will start clearing the configuration memory.

At this point, the user should be in the update-IR state in the TAP.

4. Once the Xilinx Configure instruction has been made current, the user must go from the update-IR state to the shift-DR state before the FPGA has finished clearing its configuration memory.

The approximate time it takes to clear an FPGA's configuration memory is:  $2 * 1 \text{ us} * (\# \text{ of frames per device bitstream})$ .

If the user doesn't get to the shift-dr state before  $\overline{\text{INIT}}$  goes high, then the bitstream will not be shifted into the FPGA in the right sequence and the device will not configure as expected.

5. Once  $\overline{\text{INIT}}$  has gone high, the TAP should already be in the shift-DR state.

In the shift-DR state, start shifting in the bitstream. Continue shifting in the bitstream until DONE has gone high and the startup sequence has finished.

During the time you are shifting in the bitstream via the TAP, the configuration pins LDC, HDC,  $\overline{\text{INIT}}$ ,  $\overline{\text{PROGRAM}}$ , etc. all function as they normally do during non-JTAG configuration.

Some Additional Notes:

(a) If you want to power-up the FPGA in JTAG mode, this can be done by placing a pulldown of approximately 4.7 Kohms on the  $\overline{\text{INIT}}$  pin. This pulldown has the merit of holding  $\overline{\text{INIT}}$  low to allow the user to get into JTAG, \*and\* allow the user during JTAG configuration to 'see' the  $\overline{\text{INIT}}$  pin; With the pulldown attached to  $\overline{\text{INIT}}$ , the user will see a drop of approximately 0.5V if  $\overline{\text{INIT}}$  drops low.

The alternative to using a pulldown on the  $\overline{\text{INIT}}$  pin on powerup is for the 'user' to hold  $\overline{\text{INIT}}$  low during power-up, and once the TAP is in run-test-idle, release the  $\overline{\text{INIT}}$  pin and pull it up to  $V_{CC}$ .

(b) It is possible to configure several 4K, 4KE, and/or 5K devices in a JTAG chain. But unlike non-JTAG daisy-chain configuration, this doesn't mean merging all the bitstreams into one bitstream. In the case of JTAG configuration of Xilinx devices in a JTAG chain, all devices, except the one being configured, will be placed in BYPASS mode. The one device in CONFIGURE will have its bitstream downloaded to it. After configuring this device it will be placed in BYPASS, and another device will be taken out of BYPASS into CONFIGURE.

(c) In general for the XC4000, XC4000E, and XC5200, if you are configuring these devices via JTAG, finish configuring the device first before executing any other



JTAG instructions. If the bitstream has not finished loading, then if you decide to execute some other JTAG instructions, then the configuration process via JTAG must be re-started from test-logic-reset.

(d) If boundary scan is not available after the FPGA is configured, then make sure that the release of I/Os is the last event in the startup sequence.

If boundary scan is not available, the FPGA is configured, and the I/Os are released before the startup sequence is finished, the FPGA will not respond to input signals and outputs won't respond at all.

**READBACK** – Readback through the TAP allows the user to access the readback features of the device, which would normally need to be accessed through user-specified pins. All limits of 'normal' readback are the same with readback through the TAP. Like regular readback, readback through the TAP is at a minimum of 10 KHz and at a maximum of 1 MHz. Like regular readback, the readback bistream through boundary scan has the same format.

Unlike regular readback, which can be done over and over again, readback through the TAP requires the following circuit:

1. In your schematic, or top-level synthesis design, instantiate the BSCAN and READBACK symbols.
2. Connect the BSCAN symbol pins TDI, TMS, TCK, and TDO to the boundary scan pads TDI, TMS, TCK, and TDO, respectively.
3. Next, connect the net between the TCK pad and TCK pin on the BSCAN symbol to an IBUF. Take the output of the IBUF and connect it to the CLK pin of the READBACK symbol. See [Figure 8](#).
4. After entering the above circuit, compile the design to an .lca file.
5. Make the .bit file for the .lca file by using the following option with makebits:

```
-f readclk:rdbk
```

For example, at a unix prompt:

```
% makebits -f readclk:rdbk designame
```

6. Now the FPGA is ready to perform consecutive readbacks.

READBACK is performed by loading the IR with the READBACK instruction and then shifting out the captured data from the shift-dr state in the TAP.

Perform the first readback by loading the IR with the READBACK instruction. This first readback must be finished, which means shifting out the \*entire\* readback

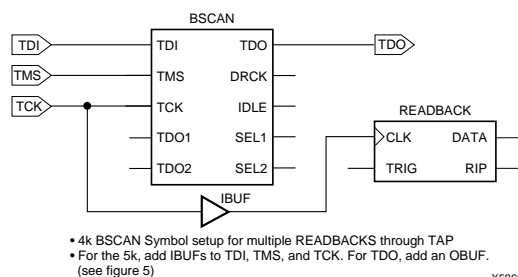
bitstream. To be safe, shift out the entire bitstream and then send three additional TCK's.

7. After performing the first readback, another readback can be performed by going to the test-logic-reset state, and re-loading the READBACK instruction and performing the READBACK as described in the previous paragraph.

In summary, consecutive readbacks are performed by starting from test-logic-reset, loading the IR with the READBACK instruction, shifting out the readback bitstream plus three additional TCK's, and then going back to the test-logic-reset state.

Alternatively, if you do not want to go back to the test-logic-reset state, realize that after shifting out readback bitstream, a minimum of 3 additional clocks are needed on the readback register. So, after doing a readback, instead of going back to test-logic-reset, a user can opt to execute some other JTAG instruction, and then perform another readback.

Also, this above procedure is only needed if you intend to do more than 1 readback. If you intend only to do a readback once, then connection between the BSCAN symbol and the READBACK symbol is not needed. In that case, all that is needed is the BSCAN symbol instantiated with the boundary scan pads (TDI, TMS, TCK, & TDO) on the top-level of the design.



**Figure 8: Symbol Setup for Multiple Readbacks**

## Boundary Scan Description Language Files

Boundary Scan Description Language (BSDL) files describe boundary-scan-capable parts in a standard format used by automated test-generation software. The order and function of bits in the boundary-scan data register are included in this description.

BSDL files are available via the Xilinx BBS (408-559-9327)

## Bibliography

The following publications contains information about the IEEE Standard 1149.1, and should be consulted for general boundary-scan information beyond the scope of this application note.

Colin M. Maunder & Rodham E. Tulloss. *The Test Access Port and Boundary Scan Architecture*. IEEE Computer Society Press, 10662 Los Vaqueros Circle, P.O. BOX 3014, Los Alamitos, CA 90720-1264.

John Fluke Mfg. Co. Inc. *The ABC of Boundary Scan Test*. John Fluke Mfg. Co. Inc., P.O. BOX 9090, Everett, WA 98206.

GenRad Inc. *Meeting the Challenge of Boundary Scan*. GenRad Inc., 300 Baker Ave., Concord, MA 01742-2174.

Ken Parker. *The Boundary Scan Handbook*. Kluwer Academic Publications, (617) 871-6600.



Headquarters	Europe	Japan
<p>Xilinx, Inc. 2100 Logic Drive San Jose, CA 95124 U.S.A.</p> <p>Tel: 1 (800) 255-7778 or 1 (408) 559-7778 Fax: 1 (800) 559-7114</p> <p>Net: hotline@xilinx.com Web: http://www.xilinx.com</p>	<p>Xilinx Sarl Jouy en Josas, France Tel: (33) 1-34-63-01-01 Net: frhelp@xilinx.com</p> <p>Xilinx GmbH Aschheim, Germany Tel: (49) 89-99-1549-01 Net: dlhelp@xilinx.com</p> <p>Xilinx, Ltd. Byfleet, United Kingdom Tel: (44) 1-932-349401 Net: ukhelp@xilinx.com</p>	<p>Xilinx, K.K. Tokyo, Japan Tel: (03) 3297-9191</p>
North America	Asia Pacific	
<p>Irvine, California (714) 727-0780</p>	<p>Xilinx Asia Pacific Hong Kong Tel: (852) 2424-5200 Net: hongkong@xilinx.com</p>	
<p>Englewood, Colorado (303)220-7541</p> <p>Sunnyvale, California (408) 245-9850</p> <p>Schaumburg, Illinois (847) 605-1972</p> <p>Nashua, New Hampshire (603) 891-1098</p> <p>Raleigh, North Carolina (919) 846-3922</p> <p>West Chester, Pennsylvania (610) 430-3300</p> <p>Dallas, Texas (214) 960-1043</p>		

© 1996 Xilinx, Inc. All rights reserved. The Xilinx name and the Xilinx logo are registered trademarks, all XC-designated products are trademarks, and the Programmable Logic Company is a service mark of Xilinx, Inc. All other trademarks and registered trademarks are the property of their respective owners.

Xilinx, Inc. does not assume any liability arising out of the application or use of any product described herein; nor does it convey any license under its patent, copyright or maskwork rights or any rights of others. Xilinx, Inc. reserves the right to make changes, at any time, in order to improve reliability, function or design and to supply the best product possible. Xilinx, Inc. cannot assume responsibility for the use of any circuitry described other than circuitry entirely embodied in its products. Products are manufactured under one or more of the following U.S. Patents: (4,847,612; 5,012,135; 4,967,107; 5,023,606; 4,940,909; 5,028,821; 4,870,302; 4,706,216; 4,758,985; 4,642,487; 4,695,740; 4,713,557; 4,750,155; 4,821,233; 4,746,822; 4,820,937; 4,783,607; 4,855,669; 5,047,710; 5,068,603; 4,855,619; 4,835,418; and 4,902,910. Xilinx, Inc. cannot assume responsibility for any circuits shown nor represent that they are free from patent infringement or of any other third party right. Xilinx, Inc. assumes no obligation to correct any errors contained herein or to advise any user of this text of any correction if such be made.