

# Module 8

## Testing of Embedded System

Lesson

41

Boundary Scan Methods  
and Standards

## Instructional Objectives

After going through this lesson the student would be able to

- Explain the meaning of the term Boundary Scan
- List the IEEE 1149 series of standards with their important features
- Describe the architecture of IEEE 1149.1 boundary scan and explain the functionality of each of its components
- Explain, with the help of an example, how a board-level design can be equipped with the boundary scan feature
- Describe the advantages and disadvantages of the boundary scan technique

## Boundary Scan Methods and Standards

### 1. Boundary Scan History and Family

Boundary Scan is a family of test methodologies aiming at resolving many test problems: from chip level to system level, from logic cores to interconnects between cores, and from digital circuits to analog or mixed-mode circuits. It is now widely accepted in industry and has been considered as an industry standard in most large IC system designs. Boundary-scan, as defined by the IEEE Std. 1149.1 standard [1-3], is an integrated method for testing interconnects on printed circuit board that is implemented at the IC level. Earlier, most Printed Circuit Board (PCB) testing was done using bed-of-nail in-circuit test equipment. Recent advances with VLSI technology now enable microprocessors and Application Specific Integrated Circuits (ASICs) to be packaged into fine pitch, high count packages. The miniaturization of device packaging, the development of surface-mounted packaging, double-sided and multi-layer board to accommodate the extra interconnects between the increased density of devices on the board reduces the physical accessibility of test points for traditional bed-of-nails in-circuit tester and poses a great challenge to test manufacturing defects in future. The long-term solution to this reduction in physical probe access was to consider building the access inside the device i.e. a boundary scan register. In 1985, a group of European companies formed Joint European Test Action Group (JETAG) and by 1988 the Joint Test Action Group (JTAG) was formed by several companies to tackle these challenges. The JTAG has developed a specification for boundary-scan testing that was standardized in 1990 by IEEE as the IEEE Std. 1149.1-1990. In 1993 a new revision to the IEEE Std. 1149.1 standard was introduced (1149.1a) and it contained many clarifications, corrections, and enhancements. In 1994, a supplement that contains a description of the boundary-scan Description Language (BSDL) was added to the standard. Since that time, this standard has been adopted by major electronics companies all over the world. Applications are found in high volume, high-end consumer products, telecommunication products, defense systems, computers, peripherals, and avionics. Now, due to its economic advantages, smaller companies that cannot afford expensive in-circuit testers are using boundary-scan. Figure 41.1 gives an overview of the boundary scan family, now known as the IEEE 1149.x standards.

Number	Description	Year
IEEE 1149.1	Testing of digital chips and interconnections between chips	Std 1149.1 – 1990
IEEE 1149.1a	Added supplement A. Rewrite of the chapter describing boundary register	Std 1149.1a – 1993
IEEE 1149.1b	Supplement B - formal description of the boundary-scan Description Language (BSDL)	Std 1149.1b – 1994
IEEE 1149.1c	Corrections, clarifications and enhancements of IEEE Std 1149.1a and Std 1149.1b. Combines 1149.1a & 1149.1b	Std 1149.1 – 2001
IEEE 1149.2	Extended Digital Serial Interface. It has merged with 1149.1 group.	Obsolete
IEEE 1149.3	Direct Access Testability Interface	Obsolete
IEEE 1149.4	Test Mixed-Signal and Analog assemblies	Std. 1149.4 – 1999
IEEE 1149.5	Standard Module Test and Maintenance (MTM) Bus Protocol. Deals with test at system level, 1149.2 has merged with.	Std. 1149.5 – 1995
IEEE 1149.6	Includes AC-coupled and/or differential nets.	Std 1149.6 - 2002
IEEE 1532	It is a derivative standard for in-system programming (ISP) of digital devices.	2000

**Fig. 41.1 IEEE 1149 Family**

The Std. 1149.1, usually referred to as the digital boundary scan, is the one that has been used widely. It can be divided into two parts: 1149.1a, or the digital Boundary Scan Standard, and 1149.1b, or the Boundary Scan Description Language (BSDL) [1,6]. Std. 1149.1 defines the chip level test architecture for digital circuits, and Std. 1149.1b is a hardware description language used to describe boundary scan architecture. The 1149.2 defines the extended digital series interface in the chip level. It has merged with 1149.1 group. The 1149.3 defines the direct access interface in contrast to 1149.2. Unfortunately this work has been discontinued. 1149.4 IEEE Standard deals with Mixed-Signal Test Bus [4]. This standard extends the test structure defined in IEEE Std. 1149.1 to allow testing and measurement of mixed-signal circuits. The standard describes the architecture and the means of control and access to analog and digital test data. The Std.1149.5 defines the bus protocol at the module level. By combining this level and Std.1149.1a one can easily carry out the testing of a PC board.

1149.6 IEEE Standard for Boundary-Scan Testing of Advanced Digital Networks is released in 2002. This standard augments 1149.1 for the testing of conventional digital networks and 1149.4 for analog networks. The 1149.6 standard defines boundary-scan structures and methods

required to test advanced digital networks that are not fully covered by IEEE Std. 1149.1, such as networks that are AC-coupled, differential, or both.

1532 IEEE Standard is developed for In-System Configuration of Programmable Devices [5]. This extension of 1149.1 standardizes programming access and methodology for programmable integrated circuit devices. Devices such as CPLDs and FPGAs, regardless of vendor, that implement this standard may be configured (written), read back, erased and verified, singly or concurrently, with a standardized set of resources based upon the algorithm description contained in the 1532 BSDL file. JTAG Technologies programming tools contain support for 1532-compliant devices and automatically generate the applications.

Clearly the testing of mixed-mode circuits at the various levels of integration will be a critical test issue for the system-on-chip design. Therefore there is a demand to combine all the boundary scan standards into an integrated one.

## 2. Boundary Scan Architecture

The boundary-scan test architecture provides a means to test interconnects between integrated circuits on a board without using physical test probes. It adds a boundary-scan cell that includes a multiplexer and latches, to each pin on the device. Figure 41.2 [1] illustrates the main elements of a universal boundary-scan device.

The Figure 41.2 shows the following elements:

- Test Access Port (TAP) with a set of four dedicated test pins: Test Data In (TDI), Test Mode Select (TMS), Test Clock (TCK), Test Data Out (TDO) and one optional test pin Test Reset (TRST\*).
- A boundary-scan cell on each device primary input and primary output pin, connected internally to form a serial boundary-scan register (Boundary Scan).
- A TAP controller with inputs TCK, TMS, and TRST\*.
- An n-bit ( $n \geq 2$ ) instruction register holding the current instruction.
- A 1-bit Bypass register (Bypass).
- An optional 32-bit Identification register capable of being loaded with a permanent device identification code.



test instructions and data are loaded from system input pins on the rising edge of TCK and driven through system output pins on its falling edge. TCK is pulsed by the equipment controlling the test and not by the tested device. It can be pulsed at any frequency (up to a maximum of some MHz). It can be even pulsed at varying rates.

- Test Data Input (TDI): an input line to allow the test instruction and test data to be loaded into the instruction register and the various test data registers, respectively.
- Test Data Output (TDO): an output line used to serially output the data from the JTAG registers to the equipment controlling the test.
- Test Mode Selector (TMS): the test control input to the TAP controller. It controls the transitions of the test interface state machine. The test operations are controlled by the sequence of 1s and 0s applied to this input. Usually this is the most important input that has to be controlled by external testers or the on-board test controller.

Test Reset Input (TRST\*): The optional TRST\* pin is used to initialize the TAP controller, that is, if the TRST\* pin is used, then the TAP controller can be asynchronously reset to a Test-Logic-Reset state when a 0 is applied at TRST\*. This pin can also be used to reset the circuit under test, however it is not recommended for this application.

## 2.2 Boundary Scan Cell

The IEEE Std. 1149.1a specifies the design of four test data registers as shown in Figure 41.2. Two mandatory test data registers, the bypass and the boundary-scan registers, must be included in any boundary scan architecture. The boundary scan register, though may be a little confusing by its name, refers to the collection of the boundary scan cells. The other registers, such as the device identification register and the design-specific test data registers, can be added optionally.

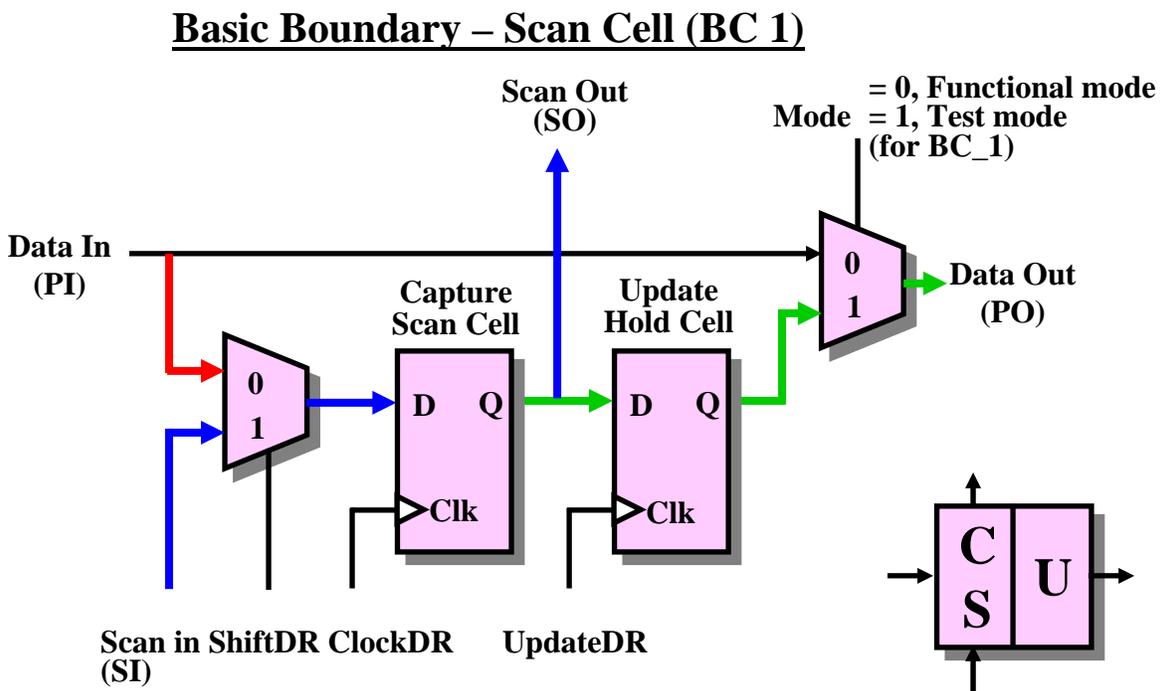


Fig. 41.3 Basic Boundary Scan Cell

Figure 41.3 [1] shows a basic universal boundary-scan cell, known as a BC\_1. The cell has four modes of operation: normal, update, capture, and serial shift. The memory elements are two D-type flip-flops with front-end and back-end multiplexing of data. It is important to note that the circuit shown in Figure 41.3 is only an example of how the requirement defined in the Standard could be realized. The IEEE 1149.1 Standard does not mandate the design of the circuit, only its functional specification. The four modes of operation are as follows:

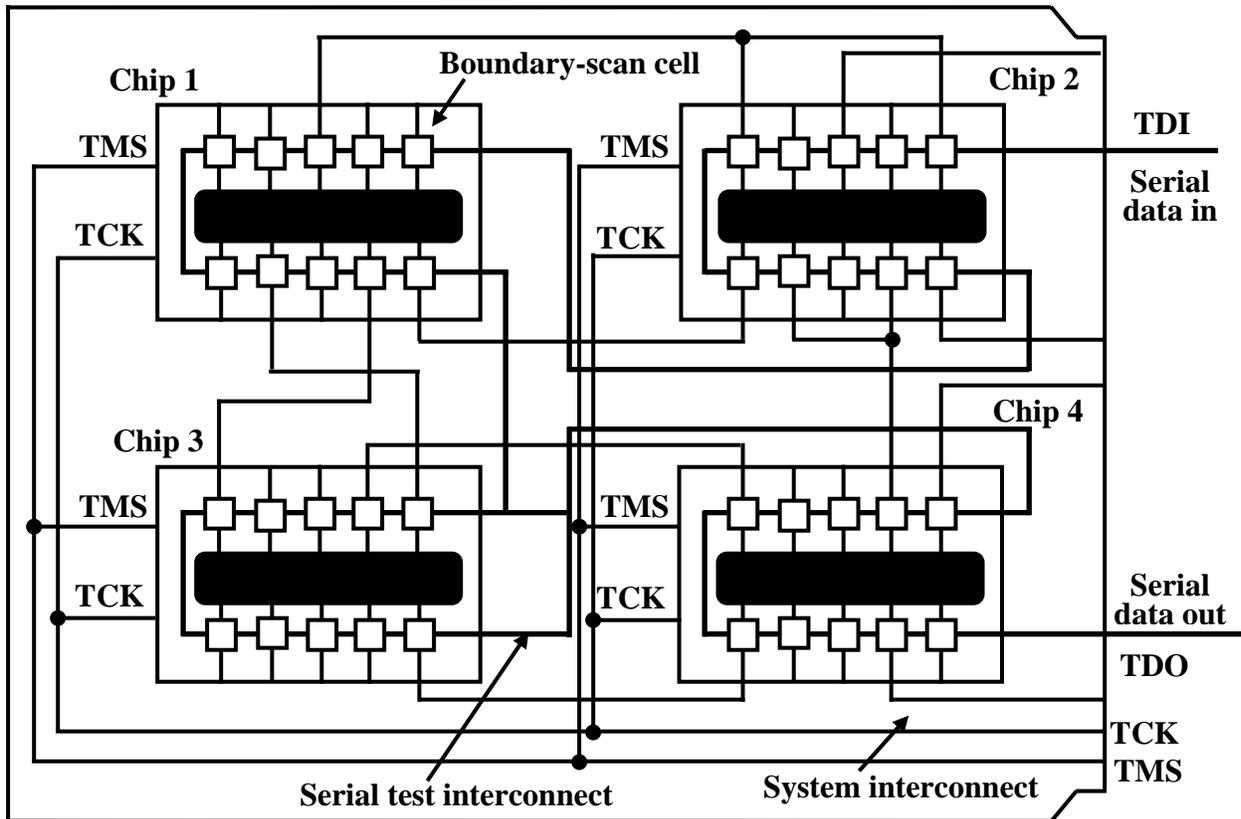
- 1) During normal mode also called serial mode, Data\_In is passed straight through to Data\_Out.
- 2) During update mode, the content of the Update Hold cell is passed through to Data\_Out. Signal values already present in the output scan cells to be passed out through the device output pins. Signal values already present in the input scan cells will be passed into the internal logic.
- 3) During capture mode, the Data\_In signal is routed to the input Capture Scan cell and the value is captured by the next ClockDR. ClockDR is a derivative of TCK. Signal values on device input pins to be loaded into input cells, and signal values passing from the internal logic to device output pins to be loaded into output cells
- 4) During shift mode, the Scan\_Out of one Capture Scan cell is passed to the Scan\_In of the next Capture Scan cell via a hard-wired path.

The Test Clock, TCK, is fed in via yet another dedicated device input pin and the various modes of operation are controlled by a dedicated Test Mode Select (TMS) serial control signal. Note that both capture and shift operations do not interfere with the normal passing of data from the parallel-in terminal to the parallel-out terminal. This allows on the fly capture of operational values and the shifting out of these values for inspection without interference. This application of the boundary-scan register has tremendous potential for real-time monitoring of the operational status of a system — a sort of electronic camera taking snapshots — and is one reason why TCK is kept separate from any system clocks.

## 2.3 Boundary Scan Path

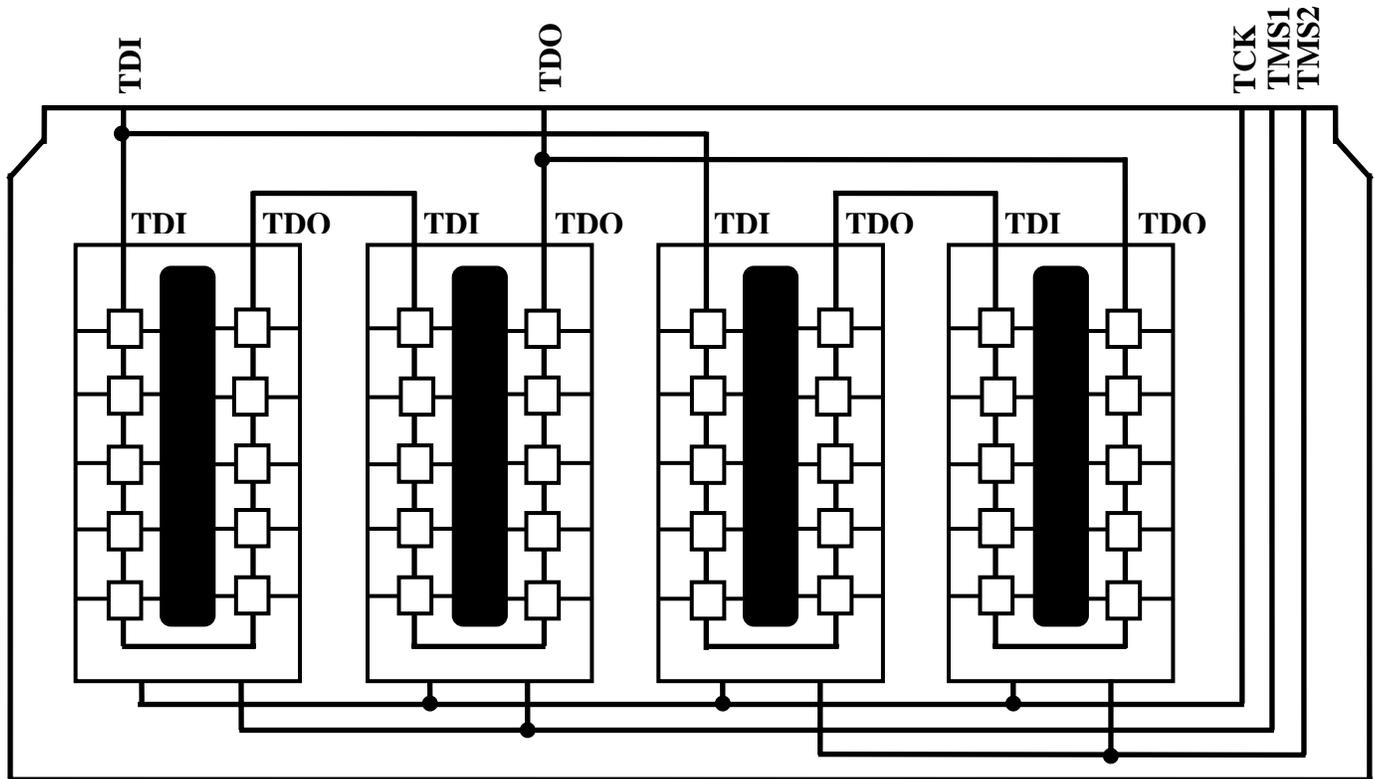
At the device level, the boundary-scan elements contribute nothing to the functionality of the internal logic. In fact, the boundary-scan path is independent of the function of the device. The value of the scan path is at the board level as shown in Figure 41.4 [1].

The figure shows a board containing four boundary-scan devices. It is seen that there is an edge-connector input called TDI connected to the TDI of the first device. TDO from the first device is permanently connected to TDI of the second device, and so on, creating a global serial scan path terminating at the edge connector output called TDO. TCK is connected in parallel to each device TCK input. TMS is connected in parallel to each device TMS input. All cell boundary data registers are serially loaded and read from this single chain.



**Fig. 41.4 MCM with Serial Boundary Scan Chain**

The advantage of this configuration is that only two pins on the PCB/MCM are needed for boundary scan data register support. The disadvantage is very long shifting sequences to deliver test patterns to each component, and to shift out test responses. This leads to expensive time on the external tester. As shown in Figure 41.5 [1], the single scan chain is broken into two parallel boundary scan chains, which share a common test clock (TCK). The extra pin overhead is one more pin. As there are two boundary scan chains, so the test patterns are half as long and test time is roughly halved. Here both chains share common TDI and TDO pins, so when the top two chips are being shifted, the bottom two chips must be disabled so that they do not drive their TDO lines. The opposite must hold true when the bottom two chips are being tested.



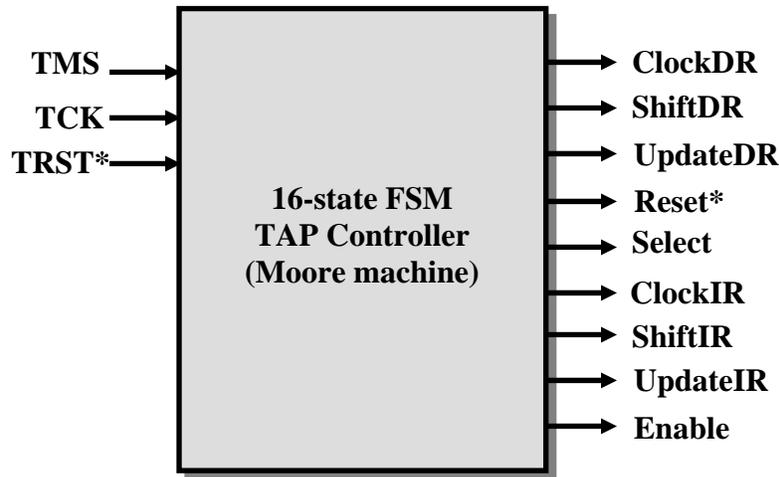
**Fig. 41.5 MCM with two parallel boundary scan chains**

## 2.4 TAP Controller

The operation of the test interface is controlled by the Test Access Port (TAP) controller. This is a 16-state finite state-machine whose state transitions are controlled by the TMS signal; the state-transition diagram is shown in Figure 41.7. The TAP controller can change state only at the rising edge of TCK and the next state is determined by the logic level of TMS. In other words, the state transition in Figure 41.6 follows the edge with label 1 when the TMS line is set to 1, otherwise the edge with label 0 is followed. The output signals of the TAP controller corresponding to a subset of the labels associated with the various states. As shown in Figure 41.2, the TAP consists of four mandatory terminals plus one optional terminal. The main functions of the TAP controller are:

- To reset the boundary scan architecture,
- To select the output of instruction or test data to shift out to TDO,
- To provide control signals to load instructions into Instruction Register,
- To provide signals to shift test data from TDI and test response to TDO, and
- To provide signals to perform test functions such as capture and application of test data.

## TAP Controller



**Fig. 41.6 Top level view of TAP Controller**

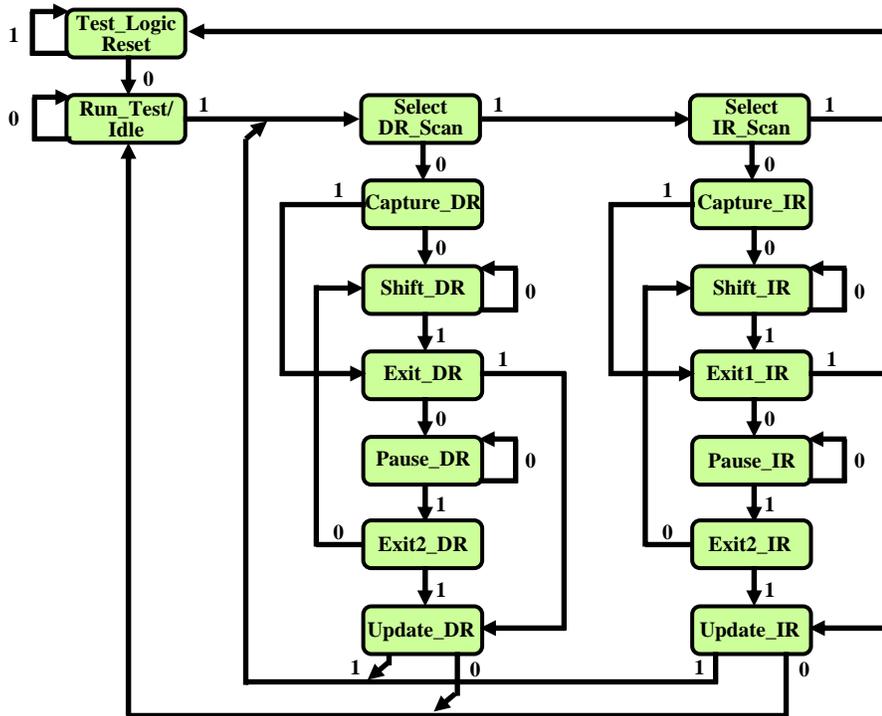
Figure 41.6 shows a top-level view of TAP Controller. TMS and TCK (and the optional TRST\*) go to a 16-state finite-state machine controller, which produces the various control signals. These signals include dedicated signals to the Instruction register (ClockIR, ShiftIR, UpdateIR) and generic signals to all data registers (ClockDR, ShiftDR, UpdateDR). The data register that actually responds is the one enabled by the conditional control signals generated at the parallel outputs of the Instruction register, according to the particular instruction.

The other signals, Reset, Select and Enable are distributed as follows:

- Reset is distributed to the Instruction register and to the target Data Register
- Select is distributed to the output multiplexer
- Enable is distributed to the output driver amplifier

It must be noted that the Standard uses the term Data Register to mean any target register except the Instruction register

## TAP Controller State Diagram



**Fig. 41.7 State transition diagram of TAP controller**

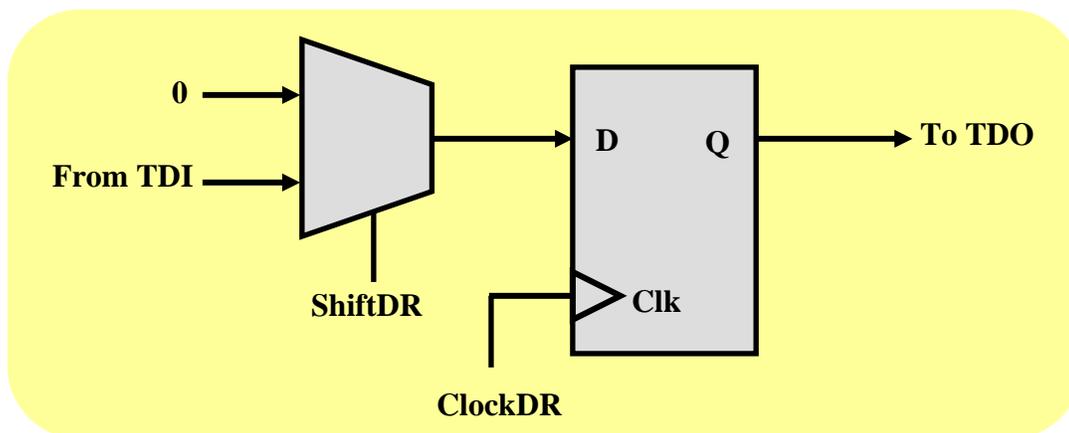
Figure 41.7 shows the 16-state state table for the TAP controller. The value on the state transition arcs is the value of TMS. A state transition occurs on the positive edge of TCK and the controller output values change on the negative edge of TCK. The 16 states can be divided into three parts. The first part contains the reset and idle states, the second and third parts control the operations of the data and instruction registers, respectively. Since the only difference between the second and the third parts are on the registers they deal with, in the following only the states in the first and second parts are described. Similar description on the second part can be applied to the third part.

1. **Test-Logic-Reset:** In this state, the boundary scan circuitry is disabled and the system is in its normal function. Whenever a Reset\* signal is applied to the BS circuit, it also goes back to this state. One should also notice that whatever state the TAP controller is at, it will go back to this state if 5 consecutive 1's are applied through TMS to the TAP controller.
2. **Run-Test/Idle:** This is a state at which the boundary scan circuitry is waiting for some test operations such as BIST operations to complete. One typical example is that if a BIST operation requires  $2^{16}$  cycles to complete, then after setting up the initial condition for the BIST operation, the TAP controller will go back to this state and wait for  $2^{16}$  cycles before it starts to shift out the test results.
3. **Select-DR-Scan:** This is a temporary state to allow the test data sequence for the selected test-data register to be initiated.

4. Capture-DR: In this state, data can be loaded in parallel to the data registers selected by the current instruction.
5. Shift-DR: In this state, test data are scanned in series through the data registers selected by the current instruction. The TAP controller may stay at this state as long as TMS=0. For each clock cycle, one data bit is shifted into (out of) the selected data register through TDI (TDO).
6. Exit-DR: All parallel-loaded (from the Capture-DR state) or shifted (from the Shift-DR state) data are held in the selected data register in this state.
7. Pause-DR: The BS pauses its function here to wait for some external operations. For example, when a long test data is to be loaded to the chip(s) under test, the external tester may need to reload the data from time to time. The Pause-DR is a state that allows the boundary scan architecture to wait for more data to shift in.
8. Exit2-DR: This state represents the end of the Pause-DR operation, allows the TAP controller to go back to ShiftDR state for more data to shift in.
9. Update-DR: The test data stored in the first stage of boundary scan cells is loaded to the second stage in this state.

## 2.5 Bypass and Identification Registers

Figure 41.8 shows a typical design for a Bypass register. It is a 1-bit register, selected by the Bypass instruction and provides a basic serial-shift function. There is no parallel output (which means that the Update\_DR control has no effect on the register), but there is a defined effect with the Capture\_DR control — the register captures a hard-wired value of logic 0.



**Fig. 41.8 Bypass register**

## 2.6 Instruction Register

As shown in Figure 41.9, an Instruction register has a shift scan section that can be connected between TDI and TDO, and a hold section that holds the current instruction. There may be some decoding logic beyond the hold section depending on the width of the register and the number of different instructions. The control signals to the Instruction register originate from the TAP controller and either cause a shift-in/shift-out through the Instruction register shift section, or cause the contents of the shift section to be passed across to the hold section (parallel Update

operation). It is also possible to load (Capture) internal hard-wired values into the shift section of the Instruction register. The Instruction register must be at least two-bits long to allow coding of the four mandatory instructions — Exttest, Bypass, Sample, Preload — but the maximum length of the Instruction register is not defined. In capture mode, the two least significant bits must capture a 01 pattern. (Note: by convention, the least-significant bit of any register connected between the device TDI and TDO pins, is always the bit closest to TDO.) The values captured into higher-order bits of the Instruction register are not defined in the Standard. One possible use of these higher-order bits is to capture an informal identification code if the optional 32-bit Identification register is not implemented. In practice, the only mandated bits for the Instruction register capture is the 01 pattern in the two least-significant bits. We will return to the value of capturing this pattern later in the tutorial.

## Instruction Register

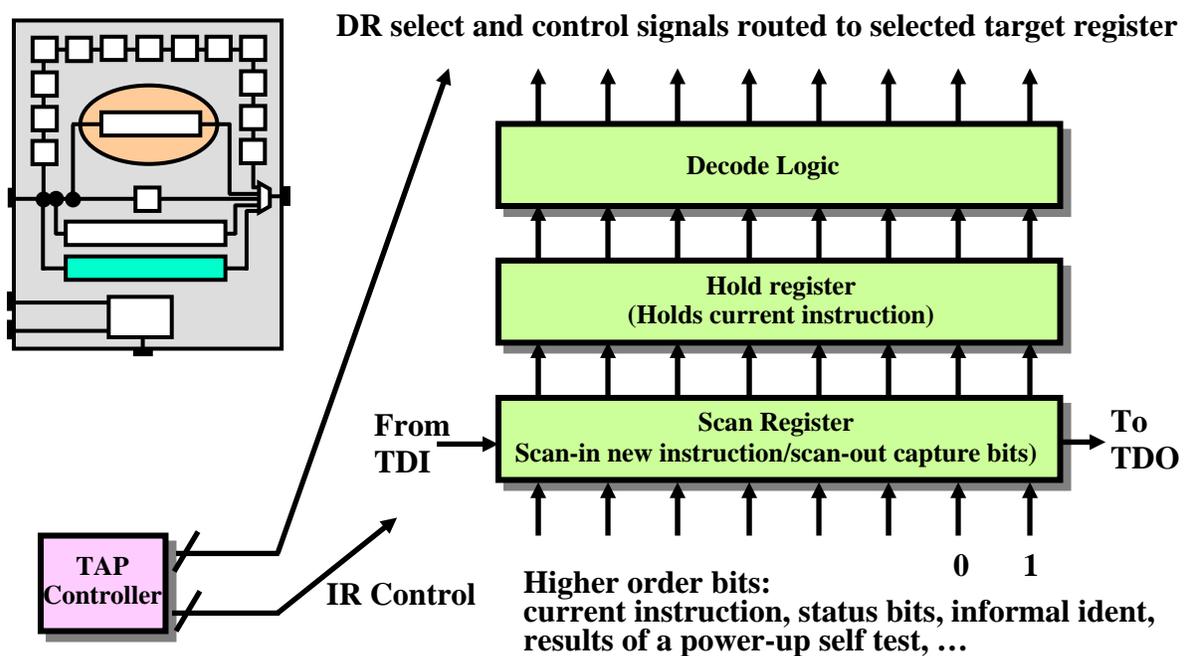


Fig. 41.9 Instruction register

## 2.7 Instruction Set

The IEEE 1149.1 Standard describes four mandatory instructions: Exttest, Bypass, Sample, and Preload, and six optional instructions: Intest, Idcode, Usercode, Runbist, Clamp and HighZ. Whenever a register is selected to become active between TDI and TDO, it is always possible to perform three operations on the register: parallel Capture followed by serial Shift followed by parallel Update. The order of these operations is fixed by the state-sequencing design of the TAP controller. For some target Data registers, some of these operations will be effectively null operations, no ops.

## Standard Instructions

Instruction	Selected Data Register
Mandatory:	
Extest	Boundary scan (formerly all-0s code)
Bypass	Bypass (initialized state, all-1s code)
Sample	Boundary scan (device in functional mode)
Preload	Boundary scan (device in function mode)
Optional:	
Intest	Boundary scan
Idcode	identification (initialized state if present)
Usercode	Identification (for PLDs)
Runbist	Result register
Clamp	Bypass (output pins in safe state)
HighZ	Bypass (output pins in high-Z state)

NB. All unused instruction codes must default to ***Bypass***

**EXTEST:** This instruction is used to test interconnect between two chips. The code for Extest used to be defined to be the all-0s code. The EXTEST instruction places an IEEE 1149.1 compliant device into an external boundary test mode and selects the boundary scan register to be connected between TDI and TDO. During this instruction, the boundary scan cells associated with outputs are preloaded with test patterns to test downstream devices. The input boundary cells are set up to capture the input data for later analysis.

**BYPASS:** A device's boundary scan chain can be skipped using the BYPASS instruction, allowing the data to pass through the bypass register. The Bypass instruction must be assigned an all-1s code and when executed, causes the Bypass register to be placed between the TDI and TDO pins. This allows efficient testing of a selected device without incurring the overhead of traversing through other devices. The BYPASS instruction allows an IEEE 1149.1 compliant device to remain in a functional mode and selects the bypass register to be connected between the TDI and TDO pins. The BYPASS instruction allows serial data to be transferred through a device from the TDI pin to the TDO pin without affecting the operation of the device.

**SAMPLE/PRELOAD:** The Sample and Preload instructions, and their predecessor the Sample/Preload instruction, selects the Boundary-Scan register when executed. The instruction sets up the boundary-scan cells either to sample (capture) values or to preload known values into the boundary-scan cells prior to some follow-on operation. During this instruction, the boundary scan register can be accessed via a data scan operation, to take a sample of the functional data entering and leaving the device. This instruction is also used to preload test data into the boundary-scan register prior to loading an EXTEST instruction.

**INTEST:** With this command the boundary scan register (BSR) is connected between the TDI and the TDO signals. The chip's internal core-logic signals are sampled and captured by the BSR cells at the entry to the "Capture\_DR" state as shown in TAP state transition diagram. The contents of the BSR register are shifted out via the TDO line at exits from the "Shift\_DR" state. As the contents of the BSR (the captured data) are shifted out, new data are sifted in at the entries to the "Shift\_DR" state. The new contents of the BSR are applied to the chip's core-logic signals during the "Update\_DR" state.

**IDCODE:** This is used to select the Identification register between TDI and TDO, preparatory to loading the internally-held 32-bit identification code and reading it out through TDO. The 32 bits are used to identify the manufacturer of the device, its part number and its version number.

**USERCODE:** This instruction selects the same 32-bit register as IDCODE, but allows an alternative 32 bits of identity data to be loaded and serially shifted out. This instruction is used for dual-personality devices, such as Complex Programmable Logic Devices and Field Programmable Gate Arrays.

**RUNBIST:** An important optional instruction is RunBist. Because of the growing importance of internal self-test structures, the behavior of RunBist is defined in the Standard. The self-test routine must be self-initializing (i.e., no external seed values are allowed), and the execution of RunBist essentially targets a self-test result register between TDI and TDO. At the end of the self-test cycle, the targeted data register holds the Pass/Fail result. With this instruction one can control the execution of the memory BIST by the TAP controller, and hence reducing the hardware overhead for the BIST controller.

**CLAMP:** Clamp is an instruction that uses boundary-scan cells to drive preset values established initially with the Preload instruction onto the outputs of devices, and then selects the Bypass register between TDI and TDO (unlike the Preload instruction which leaves the device with the boundary-scan register still selected until a new instruction is executed or the device is returned to the Test\_Logic Reset state). Clamp would be used to set up safe guarding values on the outputs of certain devices in order to avoid bus contention problems, for example.

**HIGH-Z:** It is similar to Clamp instruction, but it leaves the device output pins in a high-impedance state rather than drive fixed logic-1 or logic-0 values. HighZ also selects the Bypass register between TDI and TDO.

### 3. On Board Test Controller

So far the test architecture of boundary scan inside the chip under test has been discussed. A major problem remains is "Who is going to control the whole boundary scan test procedure?" In general there are two solutions for this problem: using an external tester and using a special on-board controller. The former is usually expensive because of the involving of an IC tester. The latter provides an economic way to complete the whole test procedure. As clear from the above description, in addition to the test data, the most important signal that a test controller has to provide is the TMS signal. There exist two methods to provide this signal in a board: the star configuration and the ring configuration as shown in Figure 41.10. In the star configuration the TMS is broadcast to all chips. Hence all chips must execute the same operation at any time. For the ring structure, the test controller provides one independent TMS signal for each chip, therefore great flexibility of the test procedure is facilitated.

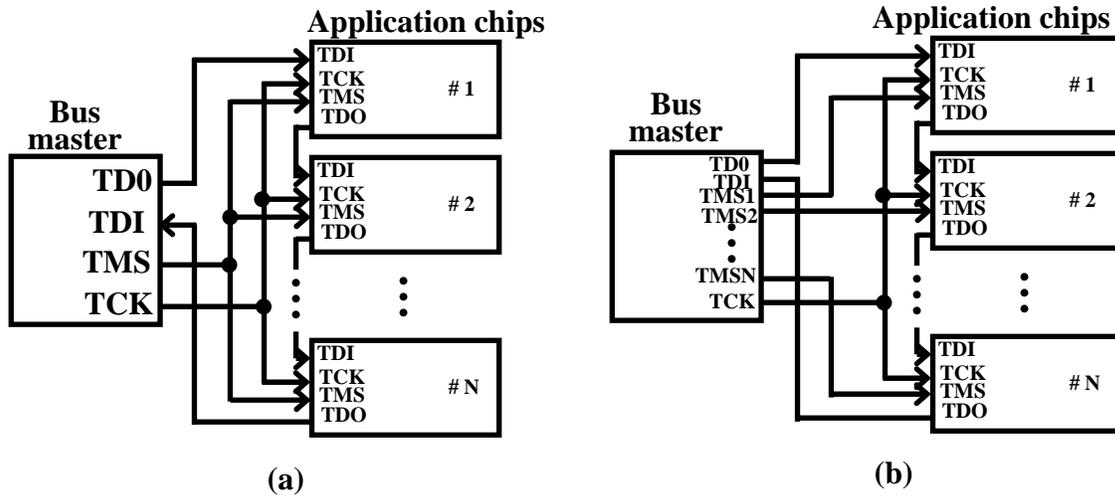
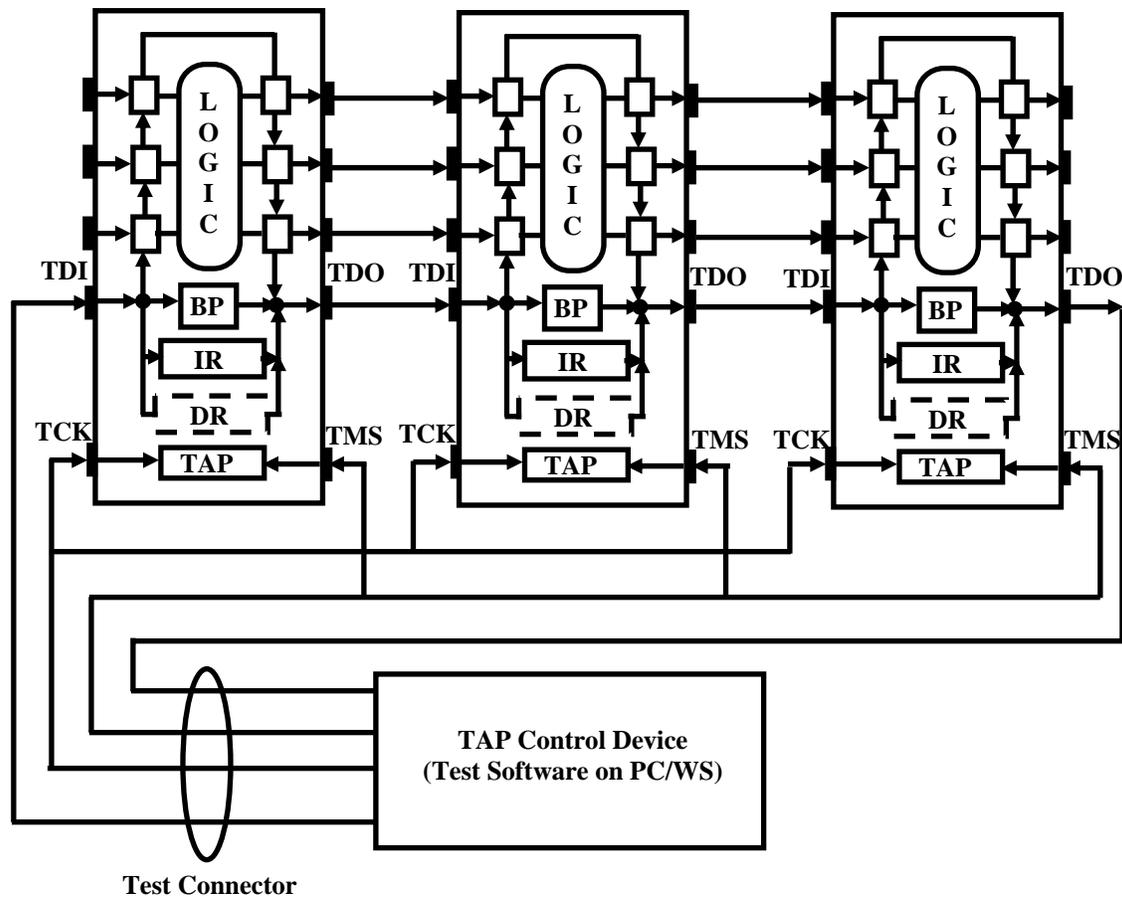


Fig. 41.10 BUS master for chips with BS: (a) star structure, (b) ring structure

#### 4. How Boundary Scan Testing Is Done

In a board design there usually can be many JTAG compliant devices. All these devices can be connected together to form a single scan chain as illustrated in Figure 41.11, "Single Boundary Scan Chain on a Board." Alternatively, multiple scan chains can be established so parallel checking of devices can be performed simultaneously.

Figure 41.11, "Single Boundary Scan Chain on a Board," illustrates the on onboard TAP controllers connected to an offboard TAP control device, such as a personal computer, through a TAP access connector. The offboard TAP control device can perform different tests during board manufacturing without the need of bed-of-nail equipment.



**Fig. 41.11 Single Boundary Scan Chain on a Board**

## 5. Simple Board Level Test Sequence

One of the first tests that should be performed for a PCB test is called the infra-structure test. This test is used to determine whether all the components are installed correctly. This test relies on the fact that the last two bits of the instruction register (IR) are always ``01". By shifting out the IR of each device in the chain, it can be determined whether the device is properly installed. This is accomplished through sequencing the TAP controller for IR read.

After the infra-structure test is successful, the board level interconnect test can begin. This is accomplished through the EXTEST command. This test can be used to check out ``opens" and ``shorts" on the PCB. The test patterns are preloaded into the output pins of the driving devices. Then they are propagated to the receiving devices and captured in the input boundary scan cells. The result can then be shifted out through the TDO pin for analysis.

These patterns can be generated and analyzed automatically, via software programs. This feature is normally offered through tools like Automatic Test Pattern Generation (ATPG) or Boundary Scan Test Pattern Generation (BTPG).

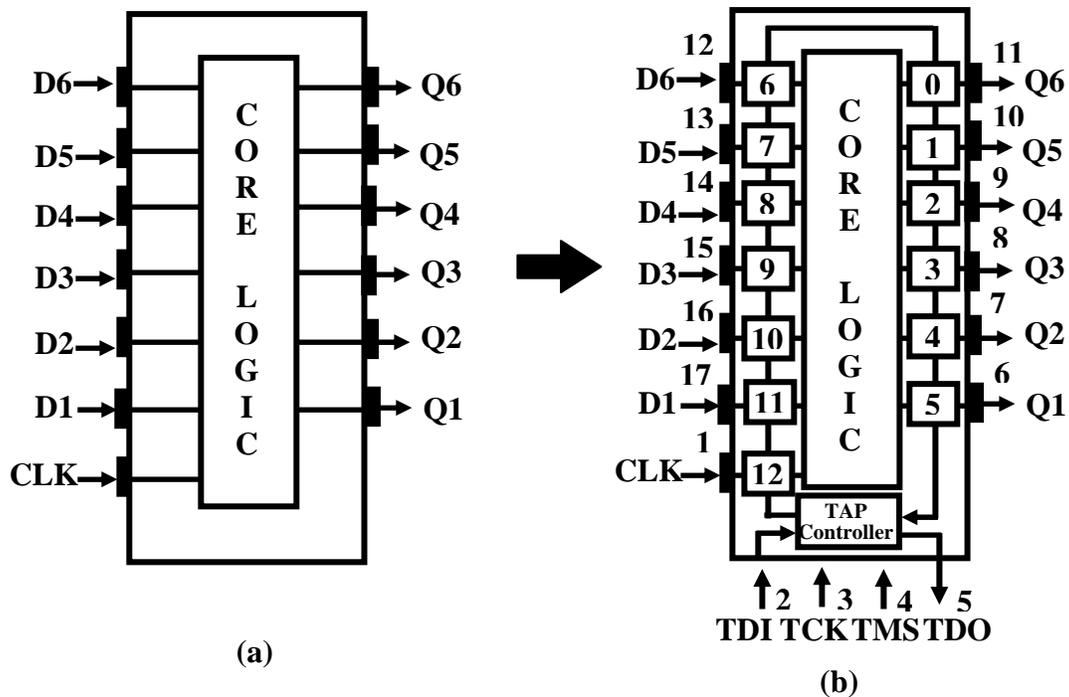
## 6. Boundary Scan Description Language

Boundary Scan Description Language (BSDL) has been approved as the IEEE Std. 1149.1b (the original boundary scan standard is IEEE Std. 1149.1a) [1,6]. This VHDL compatible

language can greatly reduce the effort to incorporate boundary scan into a chip, and hence is quite useful when a designer wishes to design boundary scan in his own style. Basically for those parts that are mandatory to the Std. 1149.1a such as the TAP controller and the BYPASS register, the designer does not need to describe them; they can be automatically generated. The designer only has to describe the specifications related to his own design such as the length of boundary scan register, the user-defined boundary scan instructions, the decoder for his own instructions, the I/O pins assignment. In general these descriptions are quite easy to prepare. In fact, currently many CAD tools already implement the boundary scan generation procedure and thus it may even not be needed for a designer to write the BSDL file: the tools can automatically generate the needed boundary scan circuitry for any circuit design as long as the I/O of the design is specified.

Any manufacturer of a JTAG compliant device must provide a BSDL file for that device. The BSDL file contains information on the function of each of the pins on the device - which are used as I/Os, power or ground. BSDL files describe the Boundary Scan architecture of a JTAG-compliant device, and are written in VHDL. The BSDL file includes:

- 1. Entity Declaration:** The entity declaration is a VHDL construct that is used to identify the name of the device that is described by the BSDL file.
- 2. Generic Parameter:** The Generic parameter specifies which package is described by the BSDL file.
- 3. Logical Port Description:** lists all of the pads on a device, and states whether that pin is an input(in bit;), output(out bit;), bidirectional (inout bit;) or unavailable for boundary scan (linkage bit;).
- 4. Package Pin Mapping:** The Package Pin Mapping shows how the pads on the device die are wired to the pins on the device package.
- 5. Use statements:** The use statement calls VHDL packages that contain attributes, types, constants, etc. that are referenced in the BSDL File.
- 6. Scan Port Identification:** The Scan Port Identification identifies the JTAG pins: TDI, TDO, TMS, TCK and TRST (if used).
- 7. TAP description:** provides additional information on the device's JTAG logic; the Instruction Register length, Instruction Opcodes, device IDCODE, etc. These characteristics are device specific.
- 8. Boundary Register description:** provides the structure of the Boundary Scan cells on the device. Each pin on a device may have up to three Boundary Scan cells, each cell consisting of a register and a latch.



**Fig. 41.12 Example to illustrate BSDL (a) core logic (b) after BS insertion**

## 7. Benefits and Penalties of Boundary Scan

The decision whether to use boundary-scan usually involves economics. Designers often hesitate to use boundary-scan due to the additional silicon involved. In many cases it may appear that the penalties outweigh the benefits for an ASIC. However, considering an analysis spanning all assembly levels and all test phases during the system's life, the benefits will usually outweigh the penalties.

### Benefits

The benefits provided by boundary-scan include the following:

- lower test generation costs
- reduced test time
- reduced time to market
- simpler and less costly testers
- compatibility with tester interfaces
- high-density packaging devices accommodation

By providing access to the scan chain I/Os, the need for physical test points on the board is eliminated or greatly reduced, leading to significant savings as a result of simpler board layouts, less costly test fixtures, reduced time on in-circuit test systems, increased use of standard interfaces, and faster time-to-market. In addition to board testing, boundary-scan allows programming almost all types of CPLDs and flash memories, regardless of size or package type, on the board, after PCB assembly. In-system programming saves money and improves throughput by reducing device handling, simplifying inventory management, and integrating the programming steps into the board production line.

## Penalties

The penalties incurred in using boundary-scan include the following:

- extra silicon due to boundary scan circuitry
- added pins
- additional design effort
- degradation in performance due to gate delays through the additional circuitry
- increased power consumption

## Boundary Scan Example

Since boundary-scan design is new to many designers, an example of gate count for a circuit with boundary scan is discussed here. This provides an estimate for the circuitry sizes required to implement the IEEE 1149.1 standard, but without the extensions defined in the standard. The example uses a library-based gate array design environment. The gate counts given are based on commercial cells and relate to a 10000 gate design in a 40-pin package. Table 1 gives the gate requirement.

Logic Element	Gate Equivalent
<b><u>Variable Size</u></b>	
Boundary-scan Register (40 cells)	680 Approx
<b><u>Fixed Sizes</u></b>	
TAP controller	131
Instruction Register (2 bits)	28
Bypass Register	9
Miscellaneous Logic	20 Approx
<b>Total</b>	<b>868 Approx</b>

Table: 1 Gate requirements for a Gate Array Boundary-scan Design

It must be noted that in Table 1 the boundary-scan implementation requires 868 gates, requiring an estimated 8 percent overhead. It also be noted that the cells used in this example were created prior to publication of the IEEE 1149.1 standard. If specific cell designs had been available to support the standard or if the vendor had placed the boundary-scan circuitry in areas of the ASIC not available to the user, then the design would have required less.

## 9. Conclusion

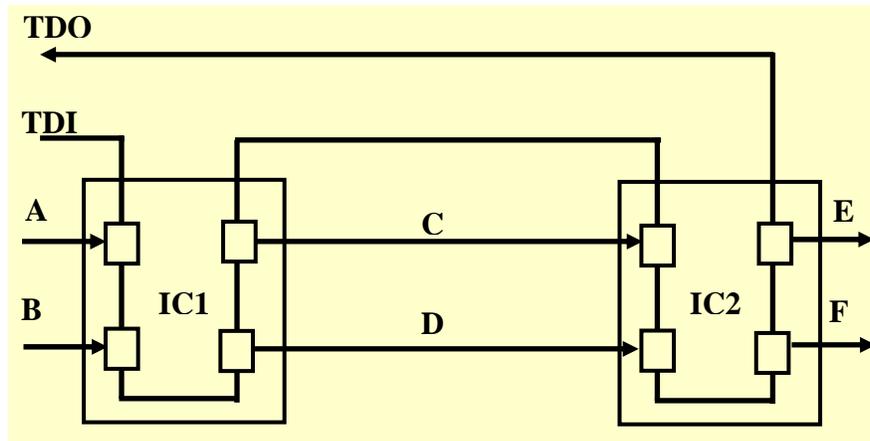
Board level testing has become more complex with the increasing use of fine pitch, high pin count devices. However with the use of boundary scan the implementation of board level testing is done more efficiently and at lower cost. This standard provides a unique opportunity to simplify the design debug and test processes by enabling a simple and standard means of automatically creating and applying tests at the device, board, and system levels. Boundary scan is the only solution for MCMs and limited-access SMT/ML boards. The standard supports external testing with an ATE. The IEEE 1532-2000 In-System Configuration (ISC) standard makes use of 1149.1 boundary-scan structures within the CPLD and FPGA devices.

## References

- [1] IEEE-SA Standards Board, 3 Park Avenue, New York, NY 10016-5997, USA, “IEEE Standard Test Access Port and Boundary-Scan Architecture”, IEEE Std 1149.1-2002, (Revision of IEEE Std 1149.1-1990), <http://grouper.ieee.org/groups/1149/1> or <http://standards.ieee.org/catalog/>
- [2] Parker, “The boundary-scan handbook: analog and digital”, Kluwer Academic Press, 1998 (2nd Edition).
- [3] M. L. Bushnell and V. D Agarwal, “Essentials of Electronic Testing” Kluwer academic Publishers, Norwell, MA, 2000.
- [4] IEEE 1149.4 Mixed-Signal Test Bus Standard web site: <http://grouper.ieee.org/groups/1149/4>
- [5] IEEE 1532 In-System Configuration Standard web site: <http://grouper.ieee.org/groups/1532/>
- [6] Agilent Technologies BSDL verification service: [http://www.agilent.com/see/bsdl\\_service](http://www.agilent.com/see/bsdl_service)

## Problems

1. What is Boundary Scan? What is the motivation of boundary scan?
2. How boundary scan technique differs from so-called bed-of-nails techniques?
3. What are the different device packaging styles?
4. What is JTAG?
5. Give an overview of the boundary scan family i.e., 1149.
6. Show boundary scan architecture and describe functions of its elements.
7. Show the basic cell of a boundary-scan register. Describe different modes of its operation.
8. A board is composed of 100 chips with 100 pins each. The length of the total scan chain is 10,000 bits. Find a possible testing strategy to reduce the scan chain length.
9. What is TAP controller? What are the main functions of TAP controller?
10. Describe a serial boundary scan chain and its operation. What are its disadvantages and discuss a strategy to overcome these.
11. Discuss different instruction sets and their functions.
12. Considering a board populated by IEEE 1149.1-compliant devices (a "pure" boundary-scan board), summarize a board-test strategy.
13. What is the goal of the infrastructure test? Is the infrastructure test mandatory or optional? Which are the main steps of an infrastructure test?
14. Consider the example depicted in the following figure.



This circuit has two primary inputs, two primary outputs and two nets that connect the ICs one to the other. There is only 1 TAP, which connects the TDI and TDO of both ICs. Prepare a test plan for this circuit.

15. Consider a board composed of 100 40-pin Boundary-Scan devices, 2,000 interconnects, an 8-bit Instruction Register per device, a 32-bit Identification Register per device, and a 10 MHz test application rate. Compute the test time to execute a test session.
16. What is BSDL. What are the different BSDL files?

Source: <http://www.nptel.ac.in/courses/Webcourse-contents/IIT%20Kharagpur/Embedded%20systems/Pdf/Lesson-41.pdf>