XAPP300 (v1.2) October 9, 2000

CoolRunner In-System Programming (ISP)

JTAG Boundary-scan and ISP

Terminology

BC	Boundary-scan Cell
BSDL	Boundary-scan Description Language
BST	Boundary-scan Test
CPLD	Complex Programmable Logic Device
IEEE	Institute of Electrical and Electronics Engineers
ISP	In-System Programmable
JEDEC	Joint Electron Device Engineering Council
JTAG	Joint Test Action Group
MC	Macrocell
PC	Personal Computer
PES	Programmer Electronic Signature
XCR3032/XCR5032	CoolRunner™ 3V/5V 32 Macrocell Device
XCR3064/PXCR5064	CoolRunner 3V/5V 64 Macrocell Device
XCR3128/XCR5128	CoolRunner 3V/5V 128 Macrocell Device
TAP	Test Access Port contained within the JTAG interface
UES	User Electronic Signature

JTAG Boundary-scan Background

JTAG Boundary-scan is the ability to test a set of devices using a standard JTAG interface instead of testing equipment which must make physical contact with the circuit pack. BST provides the ability to test the external connections of a device, the internal logic of the device, and the ability to capture data from the device during normal operation. BST provides many benefits including:

- Testability Benefits
  - No limits on the number of nets
  - Testability is designed in
  - Can force signals onto pins (Preload)
  - Can capture data from pin or core logic signals during normal operation (sample)

- Reliability Benefits
  - Not dependent on physical contact like existing test fixtures
  - Test fixtures do not deteriorate over time
  - Existing testing technologies have reached their limit
  - Better approach to handling smaller component size and components on both sides of the circuit pack
  - Shorter device interconnect routes (no fan-out required)
- Cost Benefits
  - Much cheaper test equipment
  - Reduced test preparation time
  - Reduced spare board inventories
  - Can incorporate denser technologies

ISP Background

ISP is the ability to reconfigure the logic and functionality of a device, circuit pack, or complete electronic system before, during, and after its manufacture and shipment to the end customer. ISP provides many benefits including:

- Design Benefits
  - Superior prototyping solution
  - Debug partitioning
  - Circuit pack reconfiguration during debug
- Manufacturing Benefits
  - Multi-functional hardware
  - Reconfigurability for test
  - Simplified manufacturing process
  - Reduced inventory
  - Improved manufacturing quality
- Field Support Benefits
  - No need to replace devices for new features/bug fixes
  - Possible field upgrade of hardware

JTAG Boundary-scan Features

Listed below are the JTAG Boundary-scan features implemented within Xiinx CoolRunner ISP CPLDs.

- JTAG Boundary-scan supported mainly for high pin count devices.
- Use JTAG port for JTAG Boundary-scan testing.
- JTAG Boundary-scan devices can be daisy chained together to allow multiple devices to be tested from a single JTAG stream.
- JTAG TAP Controller implementation.
- IDCODE Register contains 32 bits and a predefined format.
- Support for the following JTAG Boundary-scan instructions:
  - Sample/Preload
  - Extest
  - Bypass
- JTAG Boundary-scan Register implementation.

- TAG USERCODE Register implementation.
- Support for the following JTAG Boundary-scan instructions:
  - Idcode
  - Usercode
  - High-Z

ISP Features

Listed below are the ISP features implemented within the Xilinx CoolRunner ISP CPLDs.

- ISP supported for high pin count devices.
- Use JTAG port for ISP programming.
- No external supervoltages are required to program/erase the devices. All supervoltages are generated internal to the device from the V_{CC} input voltage.
- ISP security bit is available to protect the user program information.
- ISP devices can be daisy chained together to allow multiple devices to be programmed from a single data stream.
- ISP programming support through
  - PC Parallel Port
  - Automated Test Equipment.
  - Third party Programmers.
  - Serial Port
  - An Embedded Processor
- Number of erase/program cycles equals 1000.
- Program retention time of 20 years.
- ISP programming time less than 10 seconds for a 256 Macrocell device.
- Simultaneous programming of multiple ISP devices daisy chained together.

BST and ISP CPLD Architecture

All Xilinx high pin count CPLDs support both JTAG Boundary-scan and ISP features through a standard JTAG interface. The JTAG interface consists of two parts: a TAP Port and a TAP Controller. The TAP Port consists of four required JTAG pins (TCK, TMS, TDI, TDO) plus one optional JTAG pin (TRST*). The TAP port signals are described in the section "JTAG Interface". The TAP Controller is a sequential circuit which is used to clock in and parse JTAG instructions. The TAP Controller defined by the IEEE 1149.1 JTAG Specification is illustrated in the section, "JTAG TAP Controller."

Refer to Figure 1 for a high-level block diagram of a Xilinx ISP CPLD using the JTAG interface to access the BST and ISP functionality. As illustrated by Figure , the additional circuitry needed to implement the BST and ISP functionality consists of a TAP Controller, a JTAG Controller, an ISP Controller, an ISP Shift Register, and a Boundary-scan Register. The EEPROM array is made up of rows and columns of EEPROM cells. The ISP Shift Register contains the address to select individual EEPROM rows and contains the data which can be programmed into or read from the selected EEPROM row.

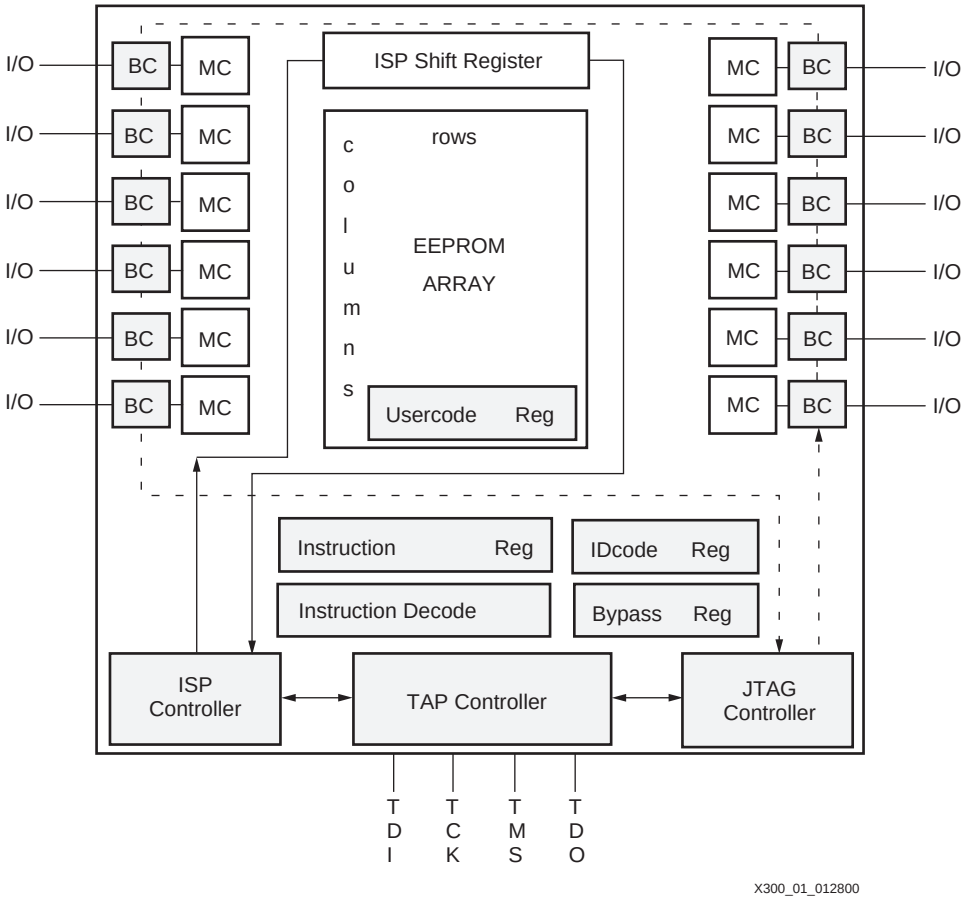

Figure 1: **ISP and BST CPLD High Level Architecture**

JTAG TAP Controller

The TAP Controller is a synchronous finite state machine that responds to changes at the TMS and TCK signals of the TAP and controls the sequence of operations of the circuitry defined by the 1149.1 IEEE Standard. This TAP Controller, as seen in Figure 2, is implemented in the Xilinx CPLDs.

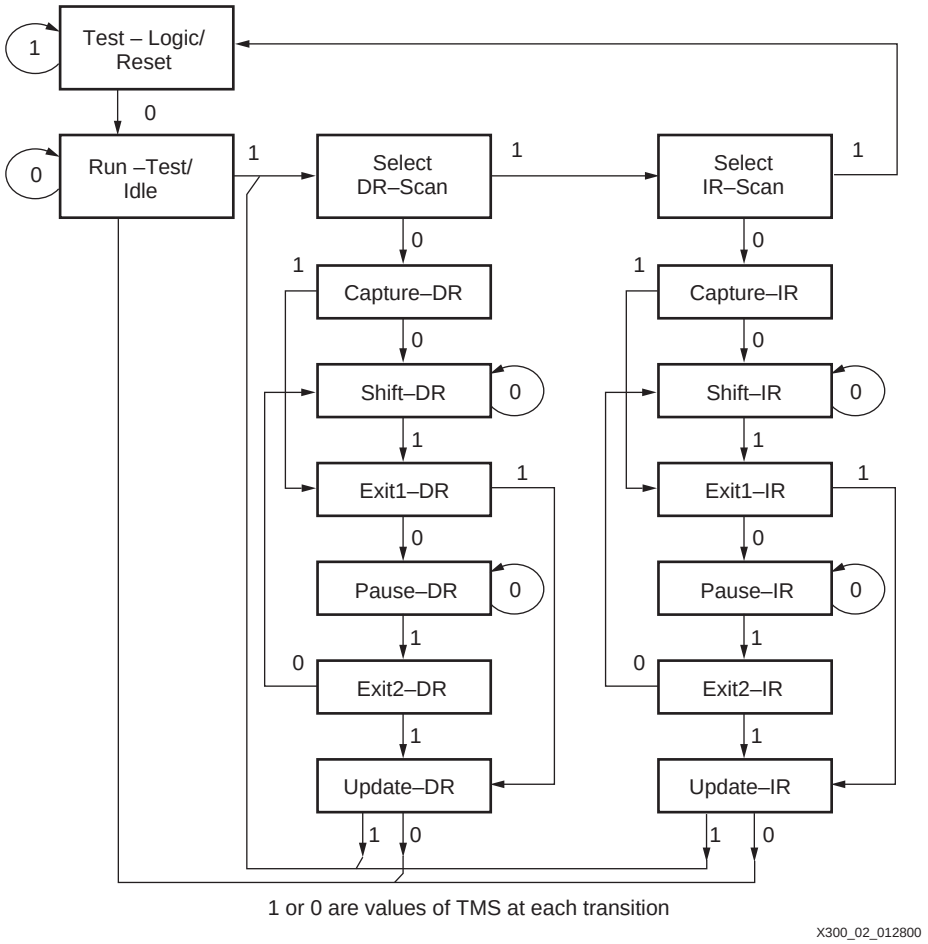

Figure 2: **JTAG TAP Controller**

The behavior of the TAP controller and other BST and ISP logic in each of the controller states is briefly described below.

Test-Logic/Reset

The BST and ISP logic is disabled so that normal operation of the on-chip system logic (i.e., in response to stimuli received through the system pins only) can continue unimpeded. This is achieved by initializing the instruction register to contain the IDCODE instruction. No matter what the original state of the controller, it will enter Test-Logic-Reset when TMS is held high for at least five rising edges of TCK. The controller remains in this state while TMS is High. Note that the TAP controller will be forced to the Test-Logic-Reset controller state at power-up.

Run-Test/Idle

All of the instructions supported by the Xilinx CPLDs do not cause functions to execute in the Run-Test/Idle controller state. Thus, all BST and ISP data registers selected by the current instruction shall retain their previous state (i.e., Idle). The instruction does not change while the TAP controller is in this state.

Select-DR-Scan

This is a temporary controller state in which all BST and ISP dataregisters selected by the current instruction retain their previous state. The instruction does not change while the TAP controller is in this state.

Select-IR-Scan

This is a temporary controller state in which all BST and ISP dataregisters selected by the current instruction retain their previous state. The instruction does not change while the TAP controller is in this state.

Capture-DR

In this controller state, data may be parallel loaded into the BST data registers selected by the current instruction on the rising edge of TCK. If a BST data register selected by the current instruction does not have parallel input, or if capturing is not required for the selected test, then the register retains its previous state. The instruction does not change while the TAP controller is in this state.

Shift-DR

In this controller state, the BST or ISP data register connected between TDI and TDO as a result of the current instruction shifts data one stage towards its serial output on each rising edge of TCK. BST or ISP data registers that are selected by the current instruction, but are not placed in the serial path, retain their previous state. The instruction does not change while the TAP controller is in this state.

Exit1-DR

This is a temporary state. All BST or ISP data registers selected by the current instruction retain their previous state. The instruction does not change while the TAP controller is in this state.

Pause-DR

This controller state allows shifting of the BST or ISP data register in the serial path between TDI and TDO to be temporarily halted. All BST or ISP data registers selected by the current instruction retain their previous state. The instruction does not change while the TAP controller is in this state.

Exit2-DR

This is a temporary state. All BST or ISP data registers selected by the current instruction retain their previous state. The instruction does not change while the TAP controller is in this state.

Update-DR

Some BST data registers may be provided with a latched parallel output to prevent changes at the parallel output while data is shifted in the associated shift-register path in response to certain instructions (e.g., EXTEST). Data is latched onto the parallel output of these BST data registers from the shift-register path on the falling edge of TCK in the Update-DR controller state. The data held at the latched parallel outputs should not change. The instruction does not change while the TAP controller is in this state.

Capture-IR

In this controller state, the shift-register contained in the instruction register loads a pattern of fixed logic values on the rising edge of TCK. In addition, design-specific data may be loaded into shift-register stages that are not required to be set to fixed values. BST or ISP data registers selected by the current instruction retain their previous state. The instruction does not change while the TAP controller is in this state.

Shift-IR

In this controller state, the shift-register contained in the instruction register is connected between TDI and TDO and shifts data one stage towards its serial output on each rising edge of TCK. BST or ISP data registers that are selected by the current instruction, but are not placed in the serial path, retain their previous state. The instruction does not change while the TAP controller is in this state.

Exit1-IR

This is a temporary state. BST or ISP data registers selected by the current instruction retain their previous state. The instruction does not change while the TAP controller is in this state and the instruction register retains its state.

Pause-IR

This controller state allows shifting of the instruction register to be temporarily halted. BST or ISP data registers selected by the current instruction retain their previous state. The instruction does not change while the TAP controller is in this state and the instruction register retains its state.

Exit2-IR

This is a temporary state. BST or ISP data registers selected by the current instruction retain their previous state. The instruction does not change while the TAP controller is in this state and the instruction register retains its state.

Update-IR

The instruction shifted into the instruction register is latched onto the parallel output from the shift-register path on the falling edge of TCK in the Update-IR controller state. Once the new instruction has been latched, it becomes the current instruction. BST or ISP data registers selected by the current instruction retain their previous state.

JTAG Registers

Figure 3 illustrates the JTAG Registers that are incorporated into Xilinx CPLDs.

Table 1 gives a description of each JTAG Register for Xilinx CPLDs.

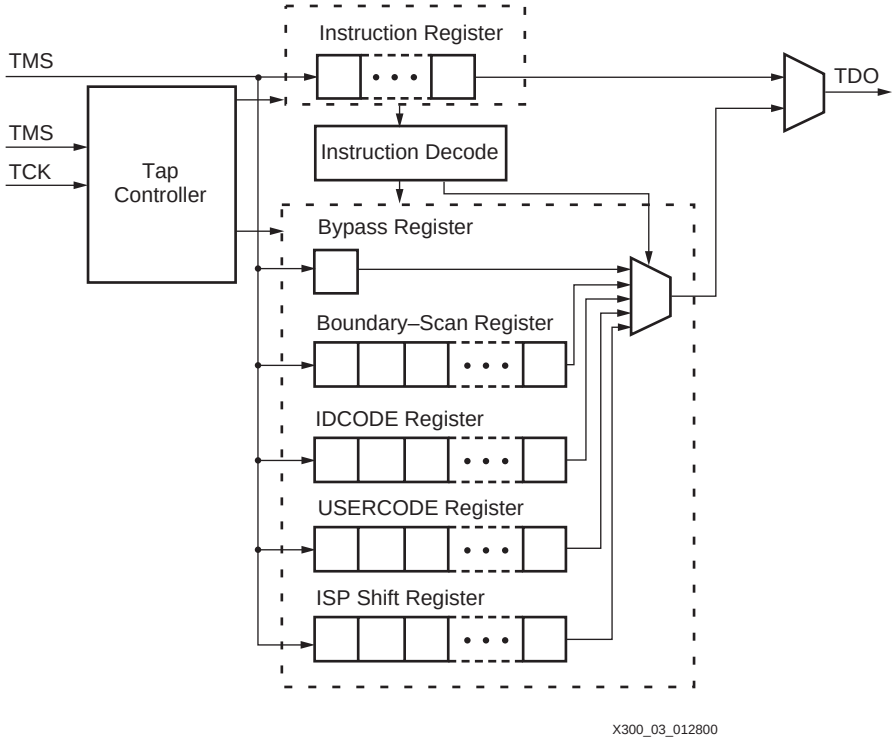

Figure 3: **JTAG Registers**

Table 1: **JTAG Register Description**

Register	# of	Register Description
Instruction Register	4	The Instruction Register is a shift-register-based design which allows an instruction to be shifted into a device. The instruction shifted into the register is latched at the completion of the shifting process. The instruction is used to select the BST or ISP operation and/or the data register to be accessed. The parallel output from the Instruction Register is latched to ensure that the BST and ISP logic is protected from the transient data patterns that will occur in its shift-register stages as new instruction data is entered.
Bypass Register	1	The Bypass Register contains a single shift-register stage and is used to provide a minimum length serial path between the TDI and TDO pins of a component when no test or program operation of that component is required. This allows more rapid movement of test/program data to and from other components on a circuit pack that are required to perform test/program operations.
Boundary-scan Register	(Device size dependent)	The Boundary-scan Register allows testing of circuitry external to the CPLD and also permits the system signals flowing into and out of the CPLD logic to be sampled and examined without causing interference with the normal operation of the CPLD logic. The Boundary-scan Register is a long shift register composed of all the Boundary-scan cells at the pins of the device.

Table 1: **JTAG Register Description**

Register	# of	Register Description
IDCODE Register	32	This register must consist of a 32-bit shift-register, parallel-in and serial out. The register contains the following information: **Bits = Usage** 0 = One predefined 1-11 = Manufacturing Identity 12-27 = Part Number 28-31 = Version
USERCODE Register	(Device size dependent)	Contains the UES information. This Register is comprised of Row 41 of the EEPROM array.
ISP Shift Register	7 Address bits, 1028 Data bits for XCR3128	Used to address the EEPROM row and contains the data that is being written into or read from the EEPROM array.

JTAG Boundary-scan Registers

The following Boundary-scan Cells were selected for Xilinx CoolRunner CPLDs dedicated input and bi-directional (I/O) pins.

An Observe-Only Dedicated Input BSR Cell

Figure 4 illustrates an Observe-only BSR Cell for a dedicated input. The advantages associated with this Boundary-scan Cell include:

- Does not require a multiplexer in the device's input speed path.
- Support for all mandatory BST instructions.
- Less silicon to implement than a BSR Cell with or without an Update Register.

Compliant BSR Cell with an Observe-only Input for Bidirectional Pin

Figure 5 illustrates a Compliant BSR Cell with an Observe-only input for a bidirectional pin. The advantages associated with this Boundary-scan Cell include:

- Does not require a multiplexer in the device's input speed path.
- Support for all mandatory BST instructions.
- Less silicon to implement than a Compliant BSR Cell without an Observe-only Input.
- The ability to control (latch) the data driving the device's output pin.
- The ability to sample and preload the output enable for the device's 3-state outputs.

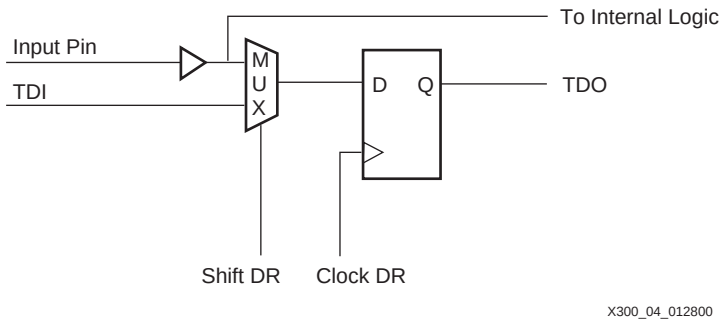

Figure 4: **An Observe-only Dedicated Input BSR Cell**

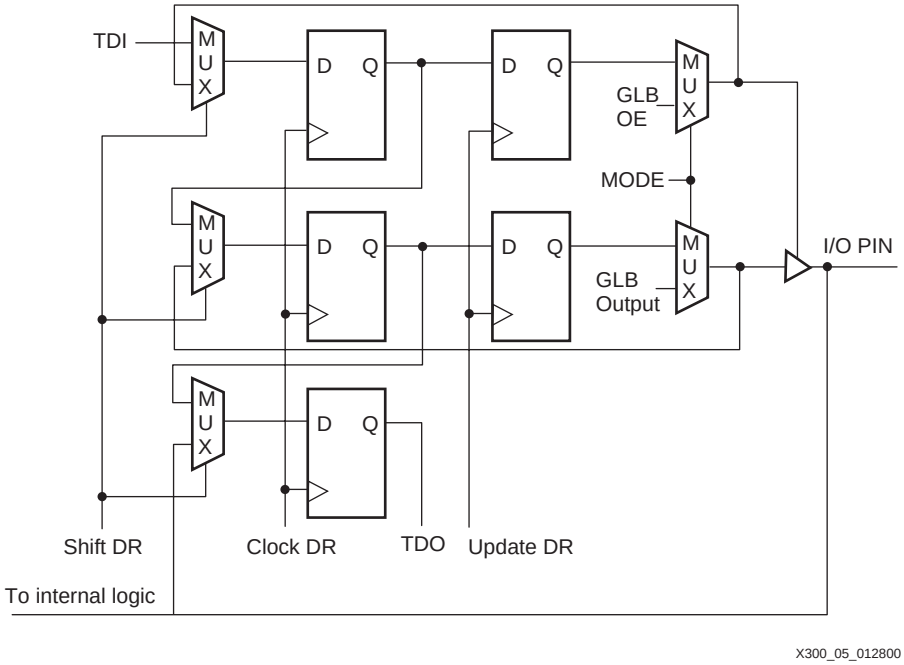

Figure 5: **Compliant BSR Cell with an Observe-only Input for a Bidirectional Pin**

JTAG Interface

As mentioned before, the JTAG pins are used to support both the BST and ISP features. Table 2 gives a description of the JTAG pins to be used to support both BST and ISP. The optional TRST* (Test Reset) signal is not required for either the BST or ISP functionality. However, leaving out the TRST* pin requires that a power up reset circuit must be designed into the device. A power-up reset circuit is included in all Xilinx CPLDs. These pins should contain an external pull-up resistor to keep the JTAG signals from floating when they are not being used.

For Xilinx CoolRunner CPLDs, an ISP ENABLE instruction is sent to the device that enables the ISP functionality (instead of using a dedicated ISP ENABLE signal). This approach is similar to the approach taken by the Altera MAX7000S family. With Xilinx' CPLDs, like the MAX700S family, the four JTAG signals use four pins which can be used for ISP or as general I/O pins if ISP is not required by the user. However the Macrocells associated with these pins can be used as buried logic when these four pins are used for ISP.

A dedicated JTAG port is used for Xilinx CPLDs containing more than 128 Macrocells.

Table 2: **JTAG Pin Description**

Pins	Name	Description
TCK	Test Clock Output	Clock pin to shift the serial data and instructions in and out of the TDI and TDO pins, respectively. TCK is also used to clock the TAP Controller state machine.
TMS	Test Mode Select	Serial input pin selects the JTAG instruction mode. TMS should be driven high during user mode operation.
TDI	Test Data Input	Serial input pin for instructions and test data. Data is shifted in on the rising edge of TCK.
TDO	Test Data Outpu	Serial output pin for instructions and test data. Data is shifted out on the falling edge of TCK. The signal is 3-stated if data is not being shifted out of the device.

JTAG Boundary-scan and ISP Commands

The TAP Controller will clock in and parse both JTAG and ISP instructions. The Xilinx ISP CPLDs contain a 4-bit Instruction Register which is large enough to contain all BST and ISP instructions. Since the Xilinx CPLDs support six JTAG Boundary-scan instructions and four ISP instructions, the Instruction Register contains four bits to represent these instructions ($2^4 = 16 > 12$). All unused instruction codes, namely 0011, 0110, 0111, 1000, 1101, and 1110 are mapped into the Boundary-scan BYPASS instruction which passes the incoming data (TDI) to the outgoing data (TDO).

The BST and ISP Commands supported by the Xilinx CPLDs are specified in the following subsections. The subsections are broken into low level and high level commands. The low level commands will be executed within the CPLD and the high level commands will be executed on a PC and/or Workstation. Please note that all high level commands are comprised of issuing a sequence of low level commands.

Low Level JTAG Boundary-scan and ISP Commands

Low Level JTAG Boundary-scan Commands

The low level JTAG Boundary-scan Commands that are supported by the Xilinx ISP CPLDs are specified in Table 3 . These are the only commands required to implement all of the required high level JTAG Boundary-scan Commands.

Low Level ISP Commands

As noted above, the low level ISP commands are basic commands which are implemented inside the CPLD. The low level ISP commands which are supported by the Xilinx ISP CPLDs are specified in Table 4. Using the ENABLE instruction before the Erase, Program, and Verify instructions allows the user to specify the outputs of the device using the JTAG Boundary-scan SAMPLE/PRELOAD command.

Please note that an ENABLE command must precede all ISP commands unless an ENABLE command has already been given for a preceding ISP command and the device has not gone through the Test-Logic/Reset TAP Controller State.

Table 3: **Supported Low Level JTAG Boundary-scan Commands**

Instruction (Instr. Code) Register Used	Description
Sample/Preload (0010) Boundary-scan Register	The mandatory SAMPLE/PRELOAD instruction allows a snapshot of the normal operation of the component to be taken and examined. It also allows data values to be loaded onto the latched parallel outputs of the Boundary-scan Shift-Register prior to selection of the other boundary-scan test instructions.
Extest (0000) Boundary-scan Register	The mandatory EXTEST instruction allows testing of off-chip circuitry and board level interconnections. Data would typically be loaded onto the latched parallel outputs of Boundary-scan Shift-Register using the Sample/Preload instruction prior to selection of the EXTEST instruction.
Bypass (1111) Bypass Register	Places the 1 bit bypass register between the TDI and TDO pins, which allows the BST data to pass synchronously through the selected device to adjacent devices during normal device operation. The Bypass instruction can be entered by holding TDI at a high value and completing an Instruction-scan cycle.

Table 3: **Supported Low Level JTAG Boundary-scan Commands**

Instruction (Instr. Code) Register Used	Description
Idcode (0001) Boundary-scan Register	elects the IDCODE register and places it between TDI and TDO, allowing the IDCODE to be serially shifted out of TDO. The IDCODE instruction permits blind interrogation of the components assembled onto a circuit pack. Thus, in circumstances where the component population may vary, it is possible to determine what components exist in a product.
STCTEST (0100) Boundary-scan Register	The STCTEST instruction is used by the Xilinx design community to check the integrity of the JTAG Boundary-scan structure. This command will be for internal Xilinx use only and will not be advertised to customers.
HighZ (0101) Bypass Register	The HIGHZ instruction places the component in a state in which al l of its system logic outputs are placed in an inactive drive state (e.g., high impedance). In this state, an in-circuit test system may drive signals onto the connections normally driven by a component output without incurring the risk of damage to the component. The HighZ instruction also forces the Bypass Register between TDI and TDO.

Table 4: **Low Level ISP Commands**

Instruction (Register Used)	Instruction Code	Description
Enable (ISP Shift Register	1001	Enables the Erase, Program, and Verify commands. Using the ENABLE instruction before the Erase, Program, and Verify instructions allows the user to specify the outputs of the device using the JTAG Boundary-scan SAMPLE/ PRELOAD command.
Erase (ISP Shift Register	1010	Erases the entire EEPROM array. The outputs during this operation can be defined by the user by using a sample preload command.
Program ISP Shift Register	1011	Programs the data in the ISP Shift Register into the addressed EEPROM row. The outputs during this operation can be defined by the user by using a sample preload command.
Verify (ISP Shift Registe	1100	Transfers the data from the addressed row to the ISP Shift Register. The data can then be shifted out and compared with the JEDEC file. The outputs during this operation can be defined by the user by using a sample preload command.

High-level JTAG Boundary-scan and ISP Commands

The High Level ISP commands are given in Table 5. Again, these commands are executed on a PC or Workstation environment and are not implemented inside the CPLD. Please note that all high level commands are comprised of issuing a sequence of low level commands.

A few examples of how high-level commands are implemented using low level commands are given in Table 6

Table 5: **High-level JTAG Boundary-scan and ISP Commands**

Instruction	Description
BULK_ERASE	Erases the entire EEPROM array.
BLANK_CHECK	Verifies that the device is erased.
PROGRAM	Programs the data into the EEPROM array.
VERIFY	Verifies that the data programmed into the EEPROM array is correct.
PR_SECURITY	Programs the security cell of the device.
RD_SECURITY	Checks to see the device is secured.
PR_USER_ID	Programs the User Signature (UES) information.
USERCODE	Reads the User Electronic Signature (UES) information.

Table 5: **High-level JTAG Boundary-scan and ISP Commands**

Instruction	Description
RD_PRGM_ID	Reads the Programmer Electronic Signature (PES) information.
PROGRAM_VERIFY	Simultaneously Programs and Verifies the EEPROM array.
BYPASS	Connects TDO to TDI with a one-half clock (TCK) cycle delay.
IDcode	Reads the IDcode information.
Pre-Condition_Outputs	Allows the user to specify the outputs during an ISP command

Table 6: **High-level Command Examples**

Command	Sequence
Program	Shift in the ENABLE instruction Shift in the PROGRAM instruction Shift in the address and data for the EEPROM row being programmed. Execute the command (Program the data into the selected EEPROM row) Repeat steps 3 and 4 until all EEPROM rows have been programmed
Verify	Shift in the ENABLE instruction Shift in the VERIFY instruction Shift in the address of the EEPROM row being verified. Execute the command (this transfers the row data into the ISP Shift Register) Shift out the data from the ISP Shift Register Compare the shifted-out data to the expected data Repeat steps 3 though 6 until all EEPROM rows have been verified
USERCODE	Shift in the ENABLE instruction Shift in a the VERIFY instruction Shift in address for Row 41 side A Execute the command (transfers the UES data into the ISP Shift Register) Shift out the data from the ISP Shift Register Shift in address for Row 41 side B Execute the command (transfers the UES data into the ISP Shift Register) Shift out the data from the ISP Shift Register
Pr_Security	Shift in the ENABLE instruction Shift in the PROGRAM instruction Shift in the address of the EEPROM row containing the security bit and shift in the data with the corresponding security bit set to the programming state and the other bits set to the non-programming state Execute the command (Program the data into the selected EEPROM row)

JTAG BSDL and ISP Chain Description File Formats

The following subsections give the BSDL and ISP Chain Description Files formats.

JTAG Boundary-scan Description Language File Format

The JTAG Boundary-scan Description Language file contains the following information:

- Mapping of Pin names to Pin Types, i.e., Inputs, Outputs, I/O, etc.
- Mapping of Pin names to physical Pin numbers.
- TAP Pin Constraints.
- OPCODES for all supported Instructions.
- Register Accessed during each Instruction.
- Boundary Register Description:
  - Sequence of Cells in Boundary Register.
  - Mapping of Cell numbers to Pin name.

To test a JTAG board, a collection of all of the BSDL files for the JTAG ICs is required along with a net list describing how these Ics are connected.

Generate an ISP File

A JEDEC file format to ISP file format program has to be written. The ISP file contains the same information as the JEDEC file but in ISP Shift Register format. The ISP file contains the following information in the following format:

- Row 0, left side data (D1027 ..... D0)
- Row 1, left side data (D1027 ..... D0)
- Row 41, left side data (D1027 ..... D0)
- Row 42, left side data (D1027 ..... D0)
- Row 0, right side data (D1027 ..... D0)
- Row 1, right side data (D1027 ..... D0)
- Row 41, right side data (D1027 ..... D0)
- Row 42, right side data (D1027 ..... D0)

ISP Chain Description File Format

This section addresses the issue of programming single or multiple ISP devices within the same JTAG chain. The JEDEC committee has put together a standard format (LtrB JC-42. 1-95-97 Chain Description File) for programming multiple devices on a single JTAG chain. A brief description of this format is given in Table 7.

It is important to recognize that both programmable and non-programmable devices can be placed on the same JTAG chain. To program a single device in a JTAG chain, the programming software must put all other devices in the JTAG chain in the JTAG Boundary Scan BYPASS mode. When in the BYPASS mode, devices pass programming data from the TDI pin to the TDO pin through a single bypass register without being affected internally, thereby enabling the programming software to erase, program, or verify the target device.

Table 7: **ISP Chain Description File Format**

Field	Description
Begin	Marks the beginning of the JC42.1 Chain Description File.
File Revision	Identifies the JC42.1 Chain Description File standard revision.
Default Declarations (Optional)	**Default Path:** Specifies a directory on a host computer of where to read and/or write data. **Default Mfr.**: Specifies the default manufacturer's code for the JEDEC chain.
Chain Site Records	Describes the device and the operation to be performed on each device in the JTAG chain. The order of the Chain Description File records must correspond to the order of the devices in the chain. In other words, there must be a one to one mapping of the device in the Chain Description File and the physical routing of the device on the circuit pack. There are two different types of device records: one for programmable devices and one for non-programmable devices. **Action Codes:** Specifies the function to be performed by the appropriate device. Examples of action codes include: program, verify, erase, bypass, etc. **Device Records**: Supports the following information: Mfr's Code, Part Name, Path Name (optional), File Name (optional), Mfr's Specific Data (optional).
End	Marks the end of the JC42.1 Chain Description File.

ISP Programming Algorithm

EEPROM technology requires a very small amount of current (60 nA) but requires a fairly lengthy programming pulse (10 ms) to program each EEPROM cell. Therefore, it is advantageous to program multiple EEPROM cells in parallel. The XCR3032/XCR5032 architecture is arranged with columns containing 255 bits of data. These 255 bits are all clocked into the device and then these 255 data bits are programmed in the EEPROM at the same time (a total of 15.3 mA). It is believed that these data bits can be clocked into the device at a maximum JTAG frequency of 10 MHz. Figure 6 illustrates the ISP Programming Flow Diagram is be used with the Xilinx CPLD architecture.

ISP Programming Times

Table 8 contains the theoretical programming time required for each Xilinx CPLD. These times were calculated using the ISP Programming Flow Diagram given in Figure 6. A conservative program cycle time of 12 ms was used to estimate the programming times for the various CPLDs. As predicted above, the total programming times are greatly dependent on the EEPROM array configuration. The more data bits that are programmed in parallel (larger data column), the fewer number of required programming pulses, and thus the shorter the total programming interval.

Please note that these programming times are better than the programming times of most CPLD vendors and that the PC or Workstation used to execute the high level commands will have a large impact on these ISP programming times.

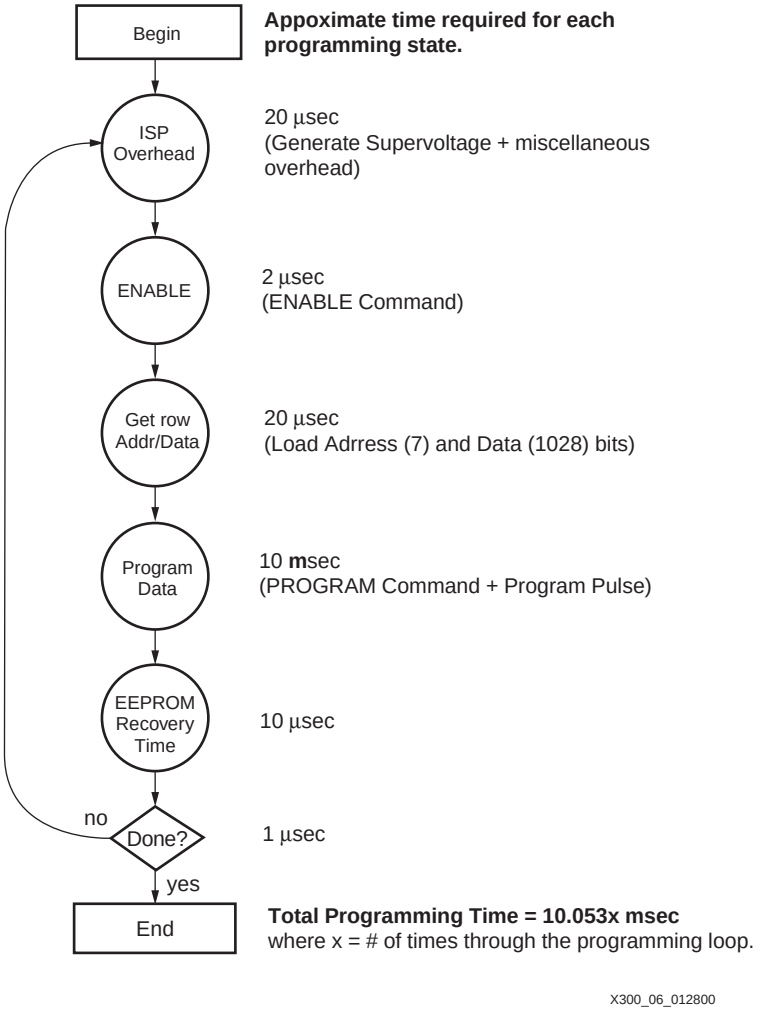

Figure 6: **XCR3128/XCR5128 ISP Programming Flow Diagram**

Table 8: **Theoretical Programming Times for Xilinx CPLDs**

Device	Width of Column Data	Theoretical Programming Times
XCR3032 / XCR5032	255	0.822 sec
XCR3064 / XCR5064	512	0.823 sec
XCR3128 / XCR5128	1024	0.824 sec

Simultaneous (Parallel) ISP Programming Algorithm

Programming multiple devices on expensive Automated Testing Equipment will take some time and will be very costly. In order to keep ISP programming costs reasonable, it is required to be able to simultaneously program multiple devices. As illustrated in the section "ISP Programming Algorithm," the number of programming pulses required to program a device or devices determines the time required to program the device(s). It is therefore advantageous to shift in data to multiple devices and then give each device a programming pulse at approximately the same time. There is approximately a 1% time increase required to

simultaneously program two devices instead of a single device. Programming two devices in series would require twice the programming time than is required to program a single device. There is approximately a 22% time increase (1.22x)[1] required to simultaneously program twenty devices instead of a single device. Programming twenty devices in series would require twenty times (20x)[1] the programming time than is required to program a single device.

In order to program two devices in parallel, the designer first selects a row address for each device, then connects the ISP Shift Register through TDI and TDO. The data is then shifted in the order that the devices appear on the chain. Next, the instruction register is selected and two PROGRAM instructions are shifted in one after another. The selected row for both devices is programmed simultaneously. This algorithm can be expanded to simultaneously program multiple devices. These devices being simultaneously programmed do not have to be of the same size or type.

Simultaneously programming multiple devices requires that a complex file be produced. This file will consist of all ISP files from the devices involved with the appropriate low-level commands intermixed.

JTAG and ISP Interfacing

A number of industry-established methods exist for JTAG/ISP interfacing with CPLDs and other integrated circuits. Xilinx CPLDs interface with the following methods:

- PC Parallel Port
- Workstation or PC Serial Port
- Embedded Processor
- Automated Test Equipment
- Third party Programmers
- High-End JTAG and ISP Tools

Boundary Scan Description Language (BSDL) descriptions of Xilinx' CPLDs are also available for use in test program development.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
02/15/00	1.1	Converted to Xilinx format.
10/09/00	1.2	Added Discontinuation Notice.