ispXP™ Configuration Usage Guidelines

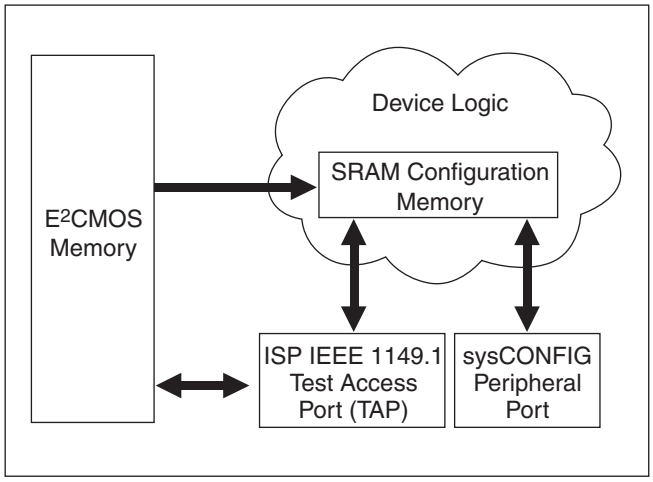
August 2002 Technical Note TN1026

1. Introduction

Traditional programmable logic devices incorporate either E²CMOS® memory or SRAM for storage of the configuration data used to define the device functionality. Each technology has its advantages and disadvantages. E²CMOS is a non-volatile technology so it does not lose its programming but it takes longer to program than SRAM and there are a limited number of times it can be programmed. SRAM is a volatile technology so it is infinitely reconfigurable but it loses its programming once power has been removed from the device and it requires an external, non-volatile source for the configuration data. On several programmable logic families of devices, Lattice Semiconductor has taken a unique approach to configuration memory by using both non-volatile E²CMOS and SRAM. Providing E²CMOS eliminates the need for external configuration devices required for SRAM only devices. SRAM gives the infinite reconfigurability not available with E²CMOS devices.

An Expanded In-System Programmability (ispXP) device uses the combination of E²CMOS non-volatile cells and SRAM technology to deliver a logic solution that provides “instant-on” at power-up, a convenient single chip solution, and the capability for infinite reconfiguration. In a Lattice device using ispXP configuration, the SRAM controls the functionality of the device. The E²CMOS memory can be thought of as the hard drive for the device, storing the configuration memory so that upon powering up the device the SRAM can be quickly configured. Figure 1 shows the architecture of an ispXP device and the relationships between the memories and the ports used to configure or program them.

Figure 1. Lattice ispXP Architecture



In this Usage Guide the terms “configuration” and “programming” are both used. “Configuration” is the act of downloading information into the SRAM memory while “programming” is the act of loading and storing data in the E²CMOS memory.

There are several ways to configure and program a Lattice device with two memory resources. This configuration guide will discuss each of the methods and will give some basic examples of how the two memories can be used together in complex applications. Some ispXP devices may vary from the ispXP Configuration guide. Please see the appendix at the end of this document for specifics on the ispXPLD and ispXPGA devices.

2. Lattice ispXP Architecture

In ispXP devices, there is an E²CMOS bit for every SRAM bit in the device. This includes any bits used for defining the functionality of the device along with bits that are included as a part of the Embedded Block RAM (EBR) mem-

ories. Having a one-to-one relationship between the SRAM and the E^2CMOS memory allows the EBR memories to be pre-initialized or to be used as Read Only Memory if needed.

2.1. Programming and Configuration Ports

As shown in Figure 1, there are two ports that can be used for configuration of the SRAM, the serial ISP™ Port or the sysCONFIG™ peripheral data port. Only the ISP Port can be used for the programming of the E^2CMOS memory. In addition to the four 1149.1 TAP pins and the twelve sysCONFIG pins, there are three common control pins that can be used in conjunction with either of the two ports. When using the ISP Port, these pins can be physically tied and are not necessary for configuring or programming the device. For the location of any of these pins on a specific device, refer to the ispXP device data sheet.

2.1.1. sysCONFIG Peripheral Port

Many embedded systems require the ability to configure devices quickly. The sysCONFIG Port can be used as an interface with a bus or microprocessor for fast configuration of the device. Eight data pins and four control pins are required to control configuration through the sysCONFIG Port. Configuration through the sysCONFIG Port is limited to the SRAM portion of an ispXP device only; there is no access to the E^2CMOS memory portion.

2.1.2. ISP Port

The ISP Port is fully compliant to the IEEE 1149.1 Test Access Port (TAP) standard, also referred to as JTAG. Lattice ispXP devices support the four-wire serial interface signals, TCK, TMS, TDI and TDO. The TAP is used to navigate the 1149.1 TAP controller, described in section 6.3. Lattice ispXP devices fully support the IEEE 1532 programming standard for programming and configuration, depending on mode selected.

Programming the E^2CMOS memory through the ISP Port can be done in any number of ways including through a PC parallel port, Automated Test Equipment (ATE) or an 1149.1 based test system. Regardless of which method is chosen, ispVM™ System software can be used to generate the necessary vectors or test files that will be used for programming. Please refer to the *ispVM System Usage Guide* associated with the ispVM System software for additional information on programming of ispXP devices.

2.2 Refresh

A third source for configuration of the SRAM memory is from the E^2CMOS memory directly. At virtually any time, the contents of the E^2CMOS memory can be downloaded into the SRAM memory. At power-up this process can be made automatic and will happen in less than 200µs. It takes only slightly more time for an ispXP device to power-up as compared to an E^2CMOS only device. An SRAM-only device with an external configuration source such as a PROM or FLASH memory may take as long as 300 ms or more to configure.

3. Pin Descriptions

The following is a pin description for programming and configuration pins used for a Lattice ispXP device. Table 1 shows a list of all pins used in sysCONFIG mode and their associated attributes. For more information on setting the Persistence Attribute, please refer to the ispLEVER help menu.

Table 1. Programming Pin Attribute List

Pin	Direction	Attribute	Characteristic	Mode Used
CFG0	Input			ISP/REFRESH/sysCONFIG
$\overline{\text{PROGRAM}}$	Input			ISP/REFRESH/sysCONFIG
DONE	Bi-directional	DONE_0D	Open-Drain or Active	ISP/REFRESH/sysCONFIG
TDI	Input		Pull-up	ISP
TCK	Input			ISP
TDO	Output		Pull-up	ISP
TMS	Input		Pull-up	ISP
$\overline{\text{INIT}}$	Bi-directional	PERSISTENT	Open-Drain	sysCONFIG
READ	Input	PERSISTENT		sysCONFIG
CCLK	Input	PERSISTENT		sysCONFIG
$\overline{\text{CS}}$	Input	PERSISTENT		sysCONFIG
DATA[0:7]	Bi-directional	PERSISTENT		sysCONFIG

3.1. Common Control Pins

Three common control pins between the sysCONFIG mode and ISP mode are always operational. The ISP mode Pins are also always active regardless of the position of the CFG0 pin. It is not recommended that ISP mode and sysCONFIG mode be used at the same time.

3.1.1. CFG0

The CFG0 pin determines if a Lattice ispXP device will configure the SRAM in sysCONFIG or ISP mode. To put the device in sysCONFIG mode, the CFG0 pin must be pulled static low. When the CFG0 pin is low, the configuration pins defined for sysCONFIG configuration shall assume their respective configuration function upon powering up or whenever the $\overline{\text{PROGRAM}}$ pin is pulsed.

To configure the SRAM using the ISP mode, CFG0 must be held high. The Persistent Pins remain in their functional (Non-configuration) state. When the CFG0 pin is held high, the SRAM memory can also be Refreshed by pulsing the $\overline{\text{PROGRAM}}$ pin. Table 2 shows a summary of the mode that is active and what memory is addressed depending on the CFG0 state. CFG0 is a dedicated pin and is not controlled by the Persistence Attribute

Table 2. Configuration Pin Selection

Memory	CFG0	Mode Implemented	Configuration/Programming Port
SRAM	0[1]	sysCONFIG	Configure from Peripheral Port
SRAM	1	ISP™ or Refresh	Configure from E^2 Cells or ISP Port
E^2	X	ISP	Program From ISP Port

1. Even though CFG0 is selecting sysCONFIG mode, the ISP Port is still active but not recommended for use.

3.1.2. DONE

The DONE pin is an active high bi-directional pin used to signify to the user that the configuration process of an ispXP device has been completed. The DONE pin is connected to the SRAM Done Bit. See section 3.4 for more information on the Done Bit which controls the DONE pin.

The DONE pin is an open-drain I/O pin. An ispXP device at the configuration initialization state will drive the DONE pin low until the device is configured successfully. The device releases the DONE pin after the CRC verification is passed in sysCONFIG mode or the Done instruction is given in ISP mode. The DONE pin can also be driven low by the host system to delay the wake up sequence. When the DONE pin is released, the device will go through the wake up process as described in section 7.

The DONE pin defaults to open-drain, but can be programmed to drive high. To program the drive-high option on the DONE pin, turn off the DONE_OD attribute associated with the DONE pin. An example of the attribute usage can be found in section 9.2. If more than one device is being programmed and the DONE_OD is turned off, it should only be turned off on the last device to be configured. Please see section 8 for more information on chaining ispXP devices for configuration. The DONE pin is a dedicated programming pin and is not controlled by the Persistence Attribute.

3.1.3. $\overline{\text{PROGRAM}}$

The $\overline{\text{PROGRAM}}$ pin is an active low control input pin. When a transition from high to low occurs, the ispXP device will prepare to be configured by the mode selected by the CFG0 pin. If CFG0 is high, the SRAM will be configured by being Refreshed from the E^2CMOS memory. If the CFG0 pin is low, the SRAM memory will prepare to be programmed by sysCONFIG mode. The $\overline{\text{PROGRAM}}$ pin must stay low for a period of t_{PRGM} in order for the device to clear the SRAM configuration as shown in Figure 3. When $\overline{\text{PROGRAM}}$ goes high, the configuration or Refresh may proceed. The $\overline{\text{PROGRAM}}$ pin is a dedicated control pin and not controlled by the persistence attribute.

3.2. sysCONFIG Control Pin Descriptions

When the device is to be configured in sysCONFIG mode, additional pins are required to support hand shaking and the byte wide bus. When not programming or configuring an ispXP device, it is desirable to have the option to have the configuration pins as extra I/O pins. A programmable bit called Persistence controls the dual-functioning configuration pins. See pin list in section 3.2 for a description of the pins controlled by the Persistence Attribute. When the Persistence Bit is programmed by setting the Persistence Attribute in ispLEVER™ software or in source code, all the configuration pins will remain as configuration pins regardless of whether the device is in configuration mode or user mode. If the Persistence Bit is not programmed, the available dual functioning configuration pins will convert to I/O pins when not configuring the ispXP. The $\overline{\text{PROGRAM}}$ pin controls the dual functional I/O pins. When the $\overline{\text{PROGRAM}}$ pin goes low in sysCONFIG mode and the Persistence Bit has not been programmed, all the configuration pins will change from I/O to programming configuration pins.

The following is a list of the control pins required for programming an ispXP device in sysCONFIG mode. Some pins have dual I/O and configuration functions depending on the programming of the persistence fuse.

3.2.1. CCLK

CCLK is an input pin in sysCONFIG mode. It is connected to the system clock for byte wide data clocking. CCLK is active in an ispXP device when the $\overline{\text{CS}}$ pin is low. Depending on the user's option, either CCLK or an internally generated clock are used to synchronize the wake up sequence of an ispXP device after configuration. Please see section 7 for a description of Wake Up. CCLK is not a dedicated configuration pin and is controlled by the Persistence Attribute.

3.2.2. $\overline{\text{CS}}$

The $\overline{\text{CS}}$ (Chip Select) pin is an active low input pin. When $\overline{\text{CS}}$ is low, it enables the DATA[0:7] data pins to receive or output a byte of data. When $\overline{\text{CS}}$ is high, DATA[0:7] data pins are tri-stated. The Persistence bit controls $\overline{\text{CS}}$ when in sysCONFIG mode. The $\overline{\text{CS}}$ is not a dedicated configuration pin and is controlled by the Persistence Attribute.

3.2.3. DATA[0:7]

The peripheral programming bus DATA[0:7] are tri-stateable bi-directional I/O pins. A byte of data is written to or read from the peripheral bus pins depending on the value of the READ signal. When the READ pin is driven high, the peripheral bus pins are output pins. When the READ pin is driven low, the peripheral bus become an output port. The pins are enabled by the $\overline{\text{CS}}$ signal. If $\overline{\text{CS}}$ is high, the peripheral bus pins are tri-stated. The peripheral bus pins DATA[0:7] are not dedicated configuration pins and are controlled by the Persistence Attribute.

3.2.4. $\overline{\text{INIT}}$

The interrupt pin $\overline{\text{INIT}}$ is an open drain control pin. It is capable of driving a low pulse out as well as detecting a low pulse driven in. The $\overline{\text{INIT}}$ pin is driven low by the Lattice ispXP device at the beginning of the configuration initialization state as an indication to the host that the device is not ready to accept the configuration data. The $\overline{\text{INIT}}$ pin should remain low as long as $\overline{\text{PROGRAM}}$ pin is low. When the device is ready to accept configuration data, the device releases the $\overline{\text{INIT}}$ pin. However, the $\overline{\text{INIT}}$ signal can be held low externally by the host system to delay con-

figuration from starting. To begin the configuration process, the $\overline{\text{INIT}}$ pin must be released by both the device and the host system. The $\overline{\text{INIT}}$ pin must then be externally pulled high to start the configuration process. During configuration, the device will drive the $\overline{\text{INIT}}$ pin low if the CRC error check fails. The $\overline{\text{INIT}}$ pin is not a dedicated configuration pin and is controlled by the Persistence Attribute.

3.2.5. READ

The READ pin is an active high control input pin. The READ pin can be held low or high prior to reading the configuration of the ispXP SRAM. On the rising edge of the READ signal, the device will to go to the READ initialization state and wait for $\overline{\text{CS}}$ to be driven low. Data will start to be read on the next rising edge of CCLK after the $\overline{\text{CS}}$ signal goes low. Data will be read from the Lattice ispXP device one byte at a time. The READ pin is not a dedicated configuration pin and is controlled by the Persistence Attribute.

3.3. ISP Program Pins

The ISP Programming pins are the standard IEEE 1149.1 TAP pins. Also referred to as JTAG pins. The ISP Pins are dedicated pins and are always accessible when the ispXP device is powered up. Three additional common control pins are also available when using the ISP Programming pins as described in section 3.1. They are not required to be used, but may be useful at times.

3.3.1. TCK

The test clock TCK provides the clock to run the TAP Controller, loading and reading the data and instruction registers. TCK can be stopped in either the high or low state and can be clocked at frequencies up to the frequency indicated in the device data sheet.

3.3.2. TDI

The Test Data Input pin is used to shift in serial test instructions and data. An internal pull-up resistor on the pin is provided.

3.3.3. TDO

The Test Data Output pin is used to shift out serial test instructions and data. When TDO is not being driven by the internal circuitry, the pin will be in a high impedance state.

3.3.4. TMS

Test Mode Select controls test operations on the TAP controller. On the falling edge of the TCK, depending on if TMS is high or low, a transition will be made in the TAP controller state machine.

3.3.5. Optional TRST

The JTAG Test Reset pin TRST is not supported in Lattice ispXP devices.

3.4. Done Bit

All Lattice ispXP devices include an internal Done Bit to indicate when programming or configuration has been completed. The Done Bit is also a requirement of the IEEE 1532 standard to indicate when programming or configuration has been successfully completed. The ispXP device supports two Done Bits, one for the SRAM configuration memory and one for the E^2CMOS memory. When the SRAM Done Bit is cleared, the device I/Os will come under the control of the Boundary Scan Registers and the DONE Pin will be driven low.

Three methods are used internally to set the SRAM Done Bit, which will set the external DONE Pin. In ISP mode, when targeting the SRAM configuration memory, the IEEE 1532 ISC_DISABLE instruction is given to set the SRAM Done Bit. In sysCONFIG mode, when the CRC is confirmed, the SRAM Done Bit will then be set. The final method to set the SRAM Done Bit is by doing a Refresh and the E^2CMOS Done Bit is downloaded to the SRAM Done Bit.

The E^2CMOS Memory Done Bit is set only by the ISC_DISABLE instruction after successfully programming and verifying the E^2CMOS memory. The E^2CMOS memory Done Bit is not connected to the DONE Pin. The only way to have the E^2CMOS memory Done Bit set the DONE Pin is to do a Refresh of the SRAM configuration memory.

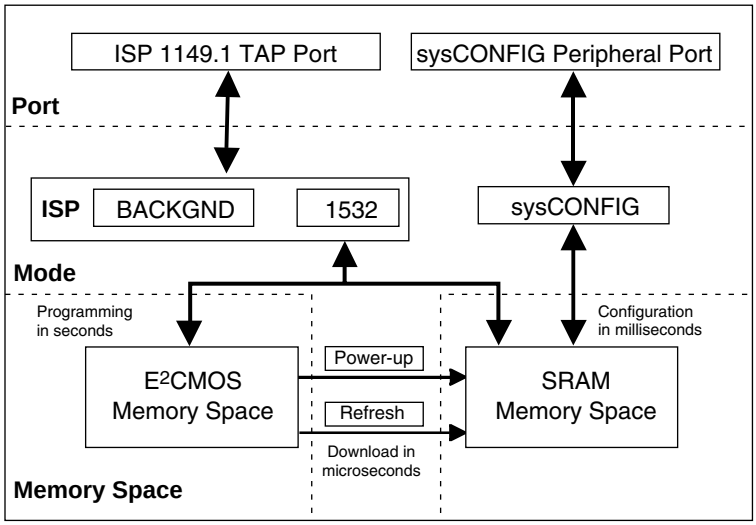
3.5. Programming and Configuration Pins Physical Description

The control pins and programming bus default to LVCMOS. The bank V_{CCO} pin determines the voltage level of the sysCONFIG pins. The JTAG pin voltage levels will be determined by the V_{CCJ} pin voltage level. Controlling the JTAG pins by V_{CCJ} allows the ispXP device to support different JTAG chain voltages. Please see *In-System Programming Design Guidelines for ispJTAG™ Devices* for further JTAG chain questions.

4. SRAM Configuration

Configuration of the SRAM can be accomplished by three different methods. Two external methods using the serial or bus port, or by downloading internally from the EE Cells directly. The E^2CMOS memory space can only be programmed by through the ISP Port. Configuration of the SRAM through the serial port is split into two modes, 1532 and Background. The sysCONFIG mode is when the SRAM is programmed directly from a peripheral device through the peripheral sysCONFIG port. The CFG0 pin controls the method of external configuration as summarized in Table 2. Figure 2 shows the major blocks in configuring a Lattice ispXP device.

Figure 2. ispXP Block Diagram



4.1. sysCONFIG Mode

When the SRAM is configured (write) or read using the sysCONFIG mode, several configuration control pins are required to read and write to the SRAM, including the clock pin CCLK, which must be provided externally to the ispXP device. With CFG0 held low, an ispXP device will be in sysCONFIG mode. In this mode, a peripheral device, such as a microprocessor will directly configure or read the SRAM through the bi-directional programming bus DATA0 through DATA7. When in sysCONFIG mode, the ISP Port is still active, but it is not recommended to use the ISP Port at the same time as the sysCONFIG Port.

4.2. ISP Mode

The SRAM of an ispXP device can be configured or read using the ISP mode. Three extra configuration control pins beyond the four standard TAP pins are required when using an ISP mode with the SRAM: CFG0, $\overline{\text{PROGRAM}}$, and DONE. With CFG0 held high, an ispXP device will be in ISP mode. The SRAM can be either configured using 1532 mode or read using Background mode. When doing any operation in ISP mode, it is recommended that the $\overline{\text{PROGRAM}}$ pin remain static.

4.3. Refresh Mode

Refresh is the internal method of configuring a Lattice ispXP device. By pulsing the $\overline{\text{PROGRAM}}$ pin low, the contents of the E^2CMOS memory will be downloaded into the SRAM. Refresh is typically done after the E^2CMOS

memory has been programmed, but can be done anytime the device E^2CMOS is not in the process of being programmed.

4.4. Power Up Mode

At the time of power up, an ispXP device can be configured by using either sysCONFIG mode or Power Up mode. By holding the CFG0 pin static high, Power Up mode downloads the contents of the E^2CMOS memory when the ispXP device achieves the appropriate power up levels. Power Up mode gives the ispXP device the advantage of the almost immediate functionality of a standard E^2CMOS CPLD by storing the configuration on board the device, eliminating the need for peripheral memory devices. If it is desired to configure an ispXP device by a peripheral device on power up then the CFG0 pin must be held static low on power up. Then follow the sysCONFIG configuration procedure for configuration.

5. sysCONFIG Configuration and Reading of the SRAM

Peripheral configuration of an ispXP device is done using sysCONFIG mode. A Lattice ispXP device supports synchronous programming of the SRAM through the bus port DATA0 to DATA7 (DATA[0:7]). Peripheral programming can come from a microprocessor, external ROM or other memory device. Additional control pins are required for sysCONFIG configuration. The following describes the steps to read and write (configure) to the SRAM portion of an ispXP device. Please see the appendix for more information on individual ispXP devices using sysCONFIG mode.

5.1. Synchronous Configuration

When in sysCONFIG mode, reading and writing the ispXP SRAM is controlled by the host device. The configuration data is provided by the host device in a byte wide stream through the peripheral sysCONFIG bus DATA0 to DATA7. The CCLK, $\overline{\text{PROGRAM}}$ and $\overline{\text{CS}}$ signals are also controlled by the host device. The following describes the read and write (configuration) cycle.

5.1.1. Synchronous Write Cycle

To write data to a Lattice ispXP device from the host device, first the SRAM must be cleared. The $\overline{\text{PROGRAM}}$ pin is toggled low for a minimum of t_{PRGM} as shown in Figure 3 to initiate clearing of the configuration SRAM. The I/Os will be tri-stated and the peripheral configuration pins take over control of the configuration process. The Done Bit will be cleared and the DONE pin will be driven low. The $\overline{\text{INIT}}$ Pin will be set low by the ispXP device to indicate that it is clearing its memory. When the $\overline{\text{INIT}}$ Pin goes high, the ispXP device is ready for programming.

The configuration process will begin when the host drives the $\overline{\text{CS}}$ low and starts CCLK. Data will be written in on the rising edge of CCLK one byte at a time. If the host needs to pause, it will drive $\overline{\text{CS}}$ high until it is ready to continue. Once the data stream is complete, the CRC will have calculated if an error occurred. If an error happened during the write process, the ispXP device will drive the $\overline{\text{INIT}}$ pin low. If no error occurred, the $\overline{\text{INIT}}$ pin will remain pulled high. After the CRC passes, the Done Bit will be set which will then drive the DONE pin high and the wake-up process will begin. Figure 3 shows the synchronous peripheral write timing sequence. The flow diagram for configuring the SRAM is shown in Figure 4.

Figure 3. sysCONFIG Write Cycle Timing Diagram

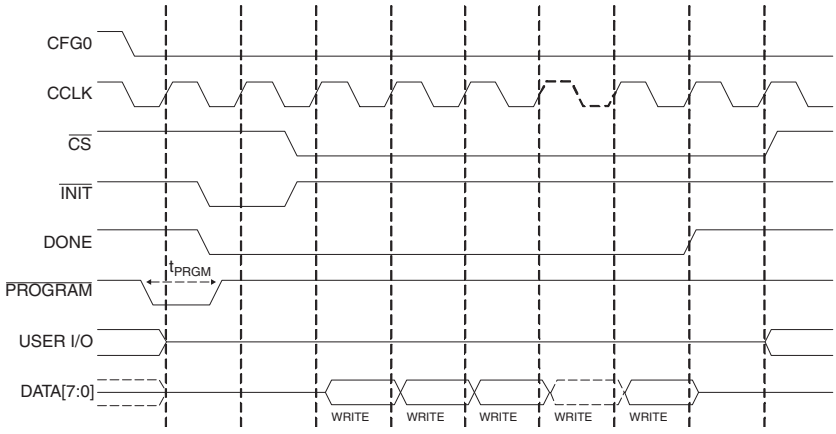
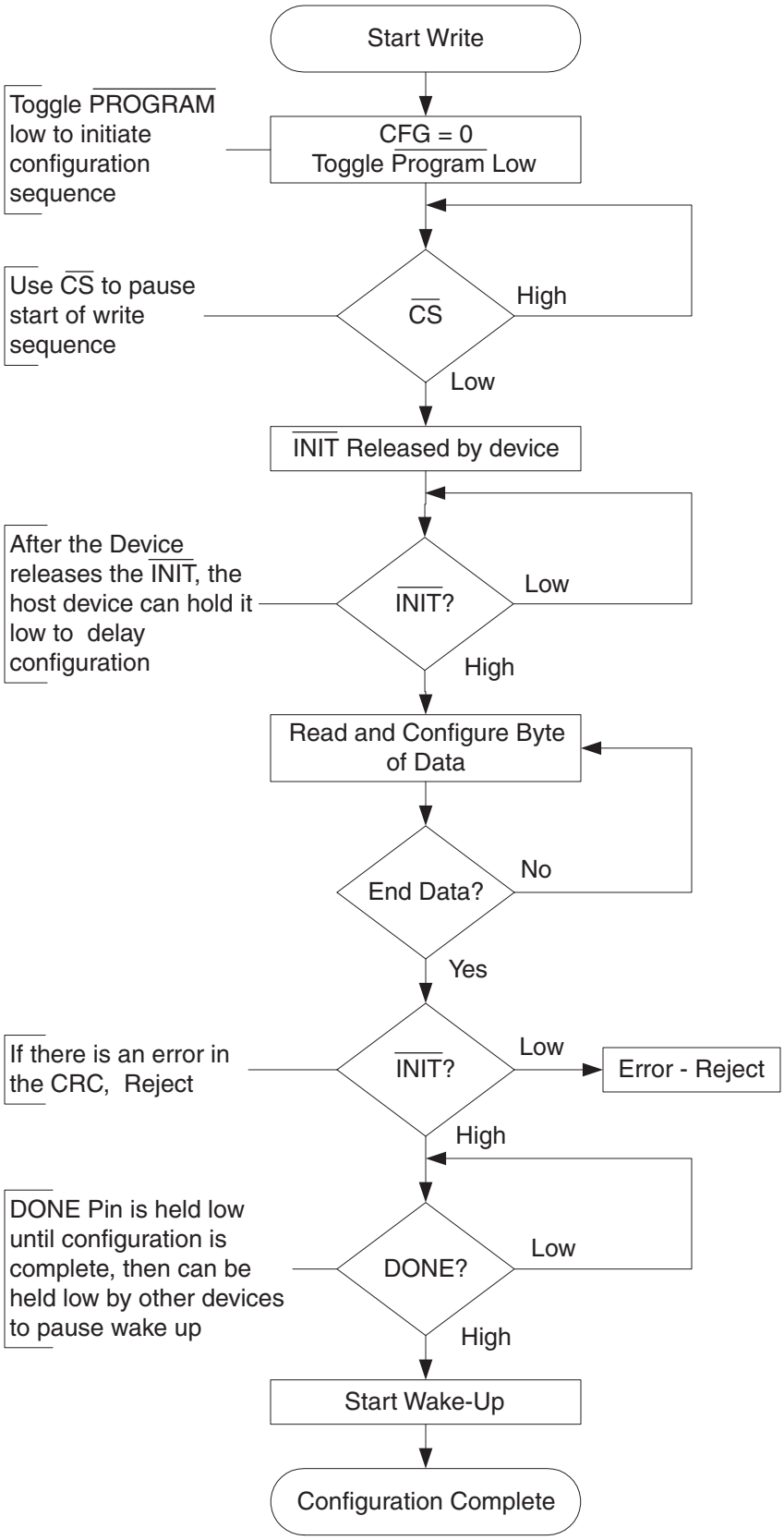


Figure 4. Synchronous Write to Configuration SRAM Flow Diagram



5.2.2. Synchronous Read Cycle

To read the configuration data from the configuration SRAM, the Persistent Attribute must have been selected to allow access to the READ, $\overline{CS}$ and DATA[0:7] pins. The READ pin must first be driven low if it is not already. To start the read cycle, drive the READ pin high. During the read option, the device will remain functional and all I/Os, except for the configuration I/Os, will remain functional. On the rising edge of the READ signal, the device will initialize the read state machine, which controls the flow of the data read out. The $\overline{CS}$ signal is used by the host to control when it is ready to accept data from the ispXP device SRAM. When the $\overline{CS}$ pin is driven low, data will be read out one byte at a time on the next rising edge of the CCLK pin. A read function is completed when the usercode is read out. Figure 5 shows synchronous peripheral read timing sequence. The flow chart for reading the SRAM is shown in Figure 6.

Figure 5. ispXP Read Timing Diagram

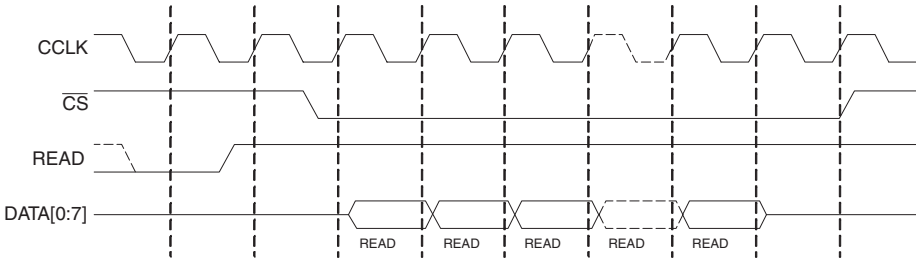
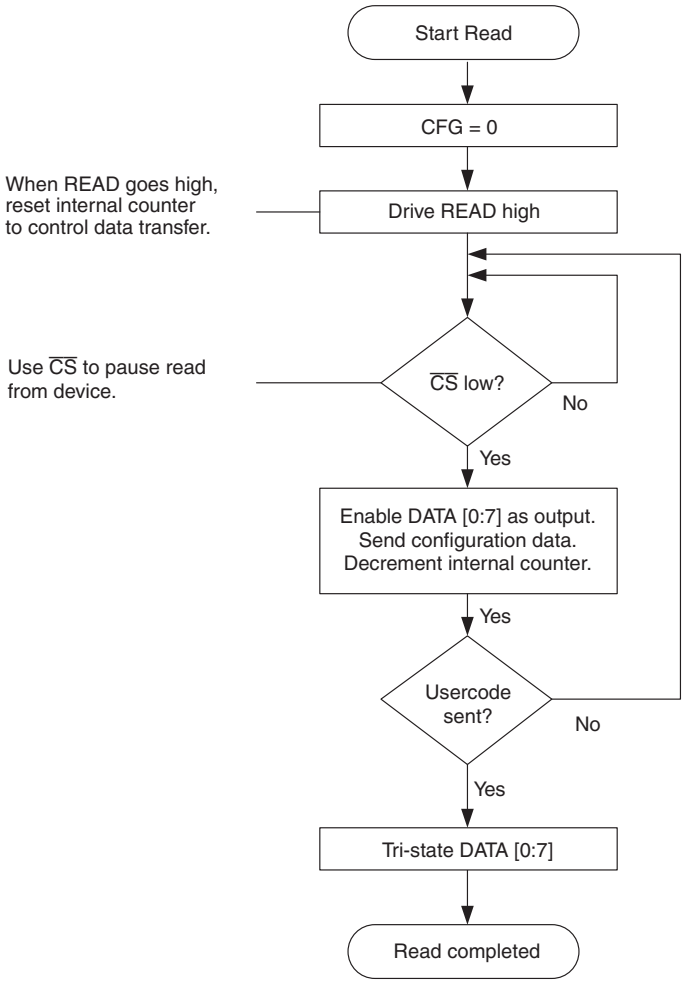


Figure 6. ispXP Read Cycle Flow Diagram



5.3. Error Detection (CRC: Cyclic Redundancy Check)

As the ispXP SRAM is being configured in sysCONFIG mode, configuration errors of the SRAM are being checked as the device is being programmed. Error detection is handled by a Cyclic Redundancy Check (CRC). As data flows through the data port, it is being feed to both the CRC circuits and the configuration SRAM. If the comparison of the CRC calculation and the CRC data that is transferred in with the configuration data is different, an error has occurred. If an error occurs, the $\overline{\text{INIT}}$ pin will be driven low to indicate to the host that there is an incorrect configuration and the ispXP device will not wake up. The Done Bit will remain cleared and will stay cleared until a proper configuration is verified by the CRC.

5.4. Power Up Configuration using sysCONFIG Mode

An ispXP device can be configured directly through the sysCONFIG Port instead of the E^2CMOS memory upon power up. By holding CFG0 static low at the time of power up, the ispXP device will be configured by the sysCONFIG Port. The I/Os will be held in a HIGHZ state and the SRAM will be reset. The $\overline{\text{INIT}}$ pin will be driven low by the ispXP device to indicate that the SRAM is in the process of being cleared. Once the SRAM is cleared, including the internal Done Bit, the DONE Pin will be driven low indicating that the device needs to be configured. Programming can now begin and should follow standard sysCONFIG configuration procedure described in section 5.1.1.

5.5. Bit Stream File Format

When configuring an ispXP device using the sysCONFIG Port, the data file format must be generated from the original design software ISC Data file. The following can be generated using the ispVM software: SRecord, Intel Hex or Standard Hex format. Table 3 describes the different sections, or frames, of the Bit Stream file.

Table 3. Configuration Bit Stream Format

Frame	Contents (D0..D7)	Description	CRC Calculation
Header	11111111	Heading	No
	1000001110100111	16-bit Preamble	No
	32 bits	32-bit trailing header	No
	1111111111111111	16-bit CRC Starter. Tells the CRC circuit to start calculating.	Yes
Data	11111110	Data Frame header	Yes
	Data bits	Number of data bits per frame depend on device	Yes
	11111111[1]	8-bit end frame	Yes
Data End	11111110	Trailer Frame Header	Yes
	32 bits	USERCODE	Yes
	1 bit	Security Bit 1 = secured	Yes
	1 bit	DONE Bit 0 = Programmed	Yes
	Fill bits	All 1's. Used to fill the rest of the frame	Yes
	11111111[1]	8-bit end frame	Yes
Trailer	11111111[1]	Stop CRC calculation instruction	Yes
	16 bits	16 bits of CRC data to compare to internally generated CRC	No
	11111111[1]	Postamble	No

1. Number of bytes is device specific.

6. ISP Mode Programming and Configuration

Programming of the E^2CMOS memory always takes place through the ISP Port. Two modes are available to access the E^2CMOS memory when in ISP mode, 1532 mode and Background mode. After re-programming the E^2CMOS memory, the SRAM can be quickly updated with the new contents of the E^2CMOS memory, minimizing the amount of time the device is inactive. The SRAM can also be addressed directly by ISP mode using the same two modes. Table 4 summarizes the different modes and if the BSCAN registers are used. Please see the appendix for more information on individual ispXP devices using ISP mode.

Table 4. Boundary Scan Support Modes

ISP Instruction	Memory	ISC 1532 Compliant	ISP Mode Used	BSCAN Register Control
Direct Program	E^2	Yes	1532	Yes
Background Program	E^2	No	Background	No
Direct Configuration	SRAM	Yes	1532	Yes
Background Read	SRAM	No	Background	No

6.1. 1532 Mode

Programming of the E^2CMOS and SRAM memory in 1532 mode follows the IEEE 1532 standard. The Boundary Scan cells take control of the I/O's during any 1532 mode instructions. The Boundary scan cells can be set to predetermined values whenever using the 1532 mode. Upon completion of programming the E^2CMOS memory, the SRAM configuration will automatically be refreshed.

6.2. Background Mode

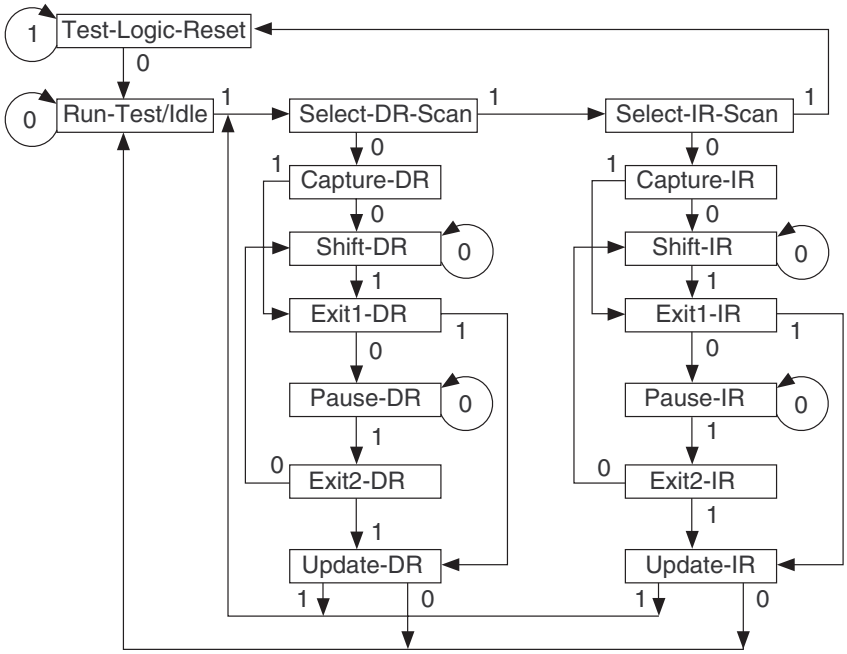
The E^2CMOS memory in an ispXP device can be programmed while the device is operational. Background programming will not interfere with the functional operation of the ispXP device. The Boundary Scan Cells will not control the I/O's making Background mode non-IEEE 1532 compliant. Once the E^2CMOS memory is programmed, the SRAM configuration can be refreshed if desired.

The SRAM can be read from in Background mode. The Boundary Scan cells will not take control of the I/Os, making an SRAM read using an ISP mode a non-IEEE 1532 compliant procedure.

6.3. JTAG TAP Controller

The purpose of the TAP controller is to control the selection and operation of the various registers and instructions that can be accessed through the TAP. It is a finite, synchronous state machine, which is controlled by TMS and is clocked from TCK. The TAP controller can be put into Test-Logic-Reset by clocking five 1's into TMS at any time. The state machine diagram for the TAP controller is shown in Figure 7. The controller is reset when it is in the Test-Logic-Reset state. A full description of the TAP controller can be found in the IEEE Std. 1149.1-1990 specification documentation

Figure 7. TAP Control State Diagram



6.4. IEEE 1149.1 Instruction Register Length

The TAP instruction register length for all Lattice ispXP devices is 8 bits. All commands are 8 bit commands. Please see the device BSDL file for specific instruction codes.

7. Wake Up

With the introduction of SRAM Based devices, new methods for coming out of configuration or programming mode have been introduced. When a device has completed configuration and the DONE Pin has been set, a Wake-Up sequence is initiated. The wake-up sequence is set by the Wake_Up Attribute, which can be set in the software tools. Several wake-up sequences are selectable by the user, as shown in Table 5. The wake-up options determine the sequence when three internal global signals and the DONE pin are activated. The three signals are the GSR (Global Set/Reset), GWD (Global Write Disable), and GOE (Global Output Enable). If the device includes on board PLLs, they can also control when a device wakes up. The attributes for Wake_On_Lock are included in the PLL section of the software configuration.

7.1. Wake Up Process

The Wake Up process is a synchronous process and as such requires a clock to which it will be synchronized. When configured in ISP or Refresh mode, the internal clock, BCLK, will control the wake-up process, which will start when the Done bit goes high as shown in Figure 8. When in sysCONFIG mode, the wake-up process is controlled by the external clock CCLK and started when the last postamble byte is read in as shown in Figure 9.

Figure 8. ISP Wake-up Timing Diagram

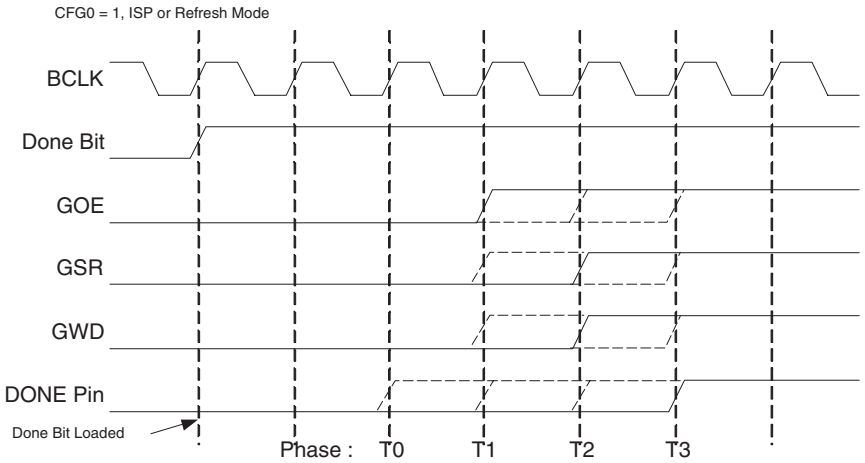
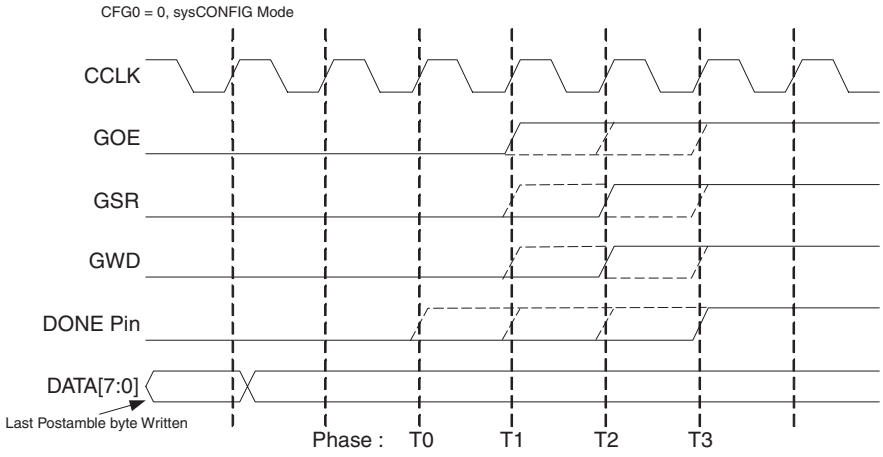


Figure 9. sysCONFIG Wake-up Timing Diagram



7.2. Selecting a Wake-Up Sequence

When selecting a Wake-Up sequence for a Lattice ispXP device, the Wake-Up signals can come up at phases T0 to T3 and will be synchronized to the appropriate clock for the mode selected. The user selects the order and the phase of wake up, and the software will set the fuses to achieve the desired wake up sequence each time the device goes through configuration. Note that in some sequences, more than one signal can be changed at a time.

If the DONE pin is selected to wake before all the other three signals, a host device can drive the DONE pin low to delay wake up until it releases the pin. If the DONE pin is not selected as the first wake signal, the ispXP device will wake regardless of what a host device does with the DONE pin. Table 5 shows the selections that can be made when waking up a ispXP device.

Table 5. User-selectable Wake-up Sequences

Sequence	Phase T0	Phase T1	Phase T2	Phase T3
1	DONE	GOE, GWD, GSR		
2	DONE		GOE, GWD, GSR	
3	DONE			GOE, GWD, GSR
4	DONE	GOE	GWD, GSR	
5	DONE	GOE		GWD, GSR
6	DONE	GOE	GWD	GSR
7	DONE	GOE	GSR	GWD
8		DONE	GOE, GWD, GSR	
9		DONE		GOE, GWD, GSR
10		DONE	GWD, GSR	GOE
11		DONE	GOE	GWD, GSR
12			DONE	GOE, GWD, GSR
13		GOE, GWD, GSR	DONE	
14		GOE	DONE	GWD, GSR
15		GOE, GWD	DONE	GSR
16		GWD	DONE	GOE, GSR
17		GWD, GSR	DONE	GOE
18		GOE, GSR	DONE	GWD
19			GOE, GWD, GSR	DONE
20		GOE, GWD, GSR		DONE
21 (Default)		**GOE**	**GWD, GSR**	**DONE**
22		GOE, GWD	GSR	DONE
23		GWD	GOE, GSR	DONE
24		GWD, GSR	GOE	DONE
25		GOE, GSR	GWD	DONE

8. Multiple ispXP Device Programming

Many systems have more than one device that needs to be configured or programmed. Using either ISP mode or sysCONFIG mode, multiple ispXP devices can be programmed. The following describes the two methods to chain together multiple devices.

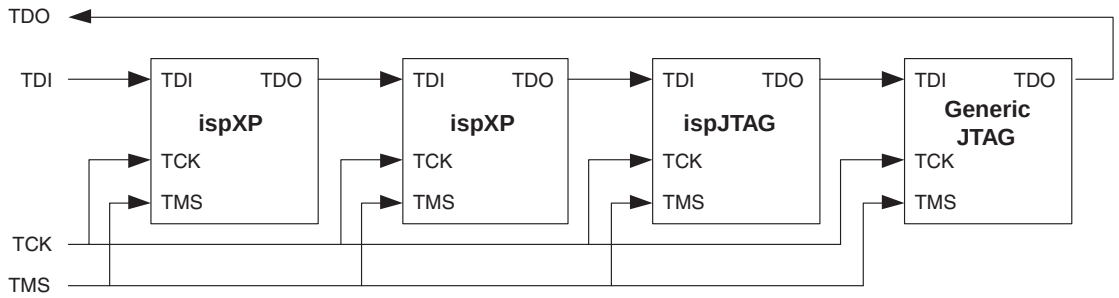
8.1. JTAG Chaining of ispXP Devices

Using ISP, it is simple to chain multiple parts together to be programmed. Figure 10 shows how multiple ispXP devices can be placed in a single JTAG chain. JTAG chains are beneficial when there are multiple types of devices to program, including In-System Programmable ispJTAG™ devices or generic JTAG devices. With the popularity of

JTAG testing, the JTAG chain provides an easy way to do testing and programming using a PC, ATE or other testing device. See *In-System Programming Design Guidelines for ispJTAG Devices* available on the Lattice web site for more information on system level design of JTAG chains.

In a JTAG chain, the DONE pin does not have to be monitored, but may be tied to other devices. If the DONE_OD option is turned off, allowing one of the devices DONE pin to drive high when its configuration is complete, it will drive all the devices DONE pins high. The drive high DONE pin would signal that all of the devices have completed configuration when it is possible the chain is not completely configured. Contention can occur between the configured device and the non-configured device DONE pins. It is recommended that if it is desired to drive high the DONE pin, the DONE_OD attribute only be set for the last ispXP device to be configured in the chain.

Figure 10. JTAG Chain



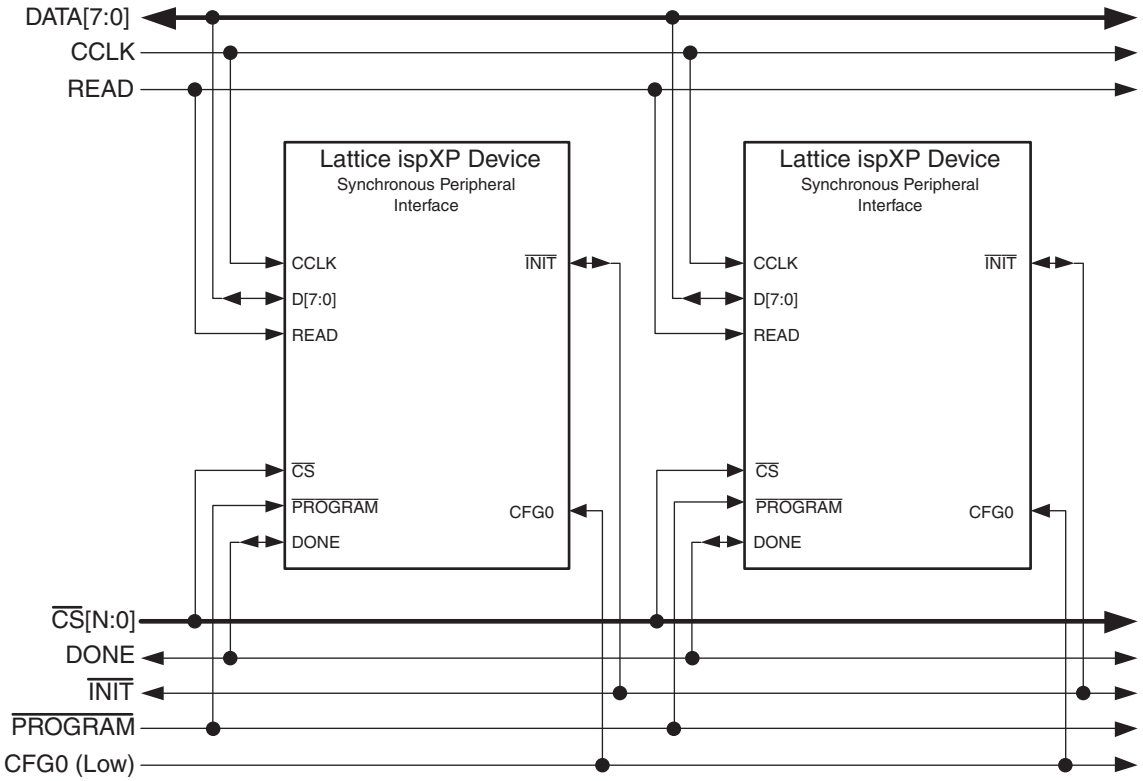
8.2. Configuration of Multiple ispXP Devices

Configuring multiple ispXP devices can be done using sysCONFIG mode. Figure 11 shows how multiple Lattice ispXP devices can be chained together for synchronous programming. Unlike the JTAG chain, where multiple vendor JTAG devices can be programmed, only ispXP devices should be chained together in sysCONFIG mode. When configuring more than one ispXP device, the chip select pin, $\overline{CS}$ should be controlled on each device with a separate chip select signal. By controlling the $\overline{CS}$ signal, each device in the chain can be selected individually.

When configuring multiple ispXP devices, the DONE pin is usually left as an open-drain. The option is available to set any of the DONE pin's open-drain characteristics to a drive- high capability. It is recommended that only the last device in the chain being configured use the drive-high DONE pin if drive high is needed. By having one device drive high, it will force all the DONE pins high and could possibly cause contention if another device has not released its DONE pin yet.

When more than one device is chained together, the DONE pin and $\overline{INIT}$ pin should have external resistors to help pull up the open drain signals. The weak internal pull-up is not strong enough to support many devices in parallel. For the DONE pin, the external resistor is recommended when all the devices DONE pin remains open drain.

Figure 11. Peripheral Programming of ispXP



9. Examples of VHDL/Verilog and Synplicity/Exemplar Attributes

The following are examples of the attributes that control wake up, Persistence, and other attributes associated with configuring an ispXP device. Selections can also be made using the Constraint Manager in ispLEVER Software as well.

9.1. PLL Wake On Lock Attribute (Simple PLL Example)

9.1.1. Verilog, Simple PLL Declaration

```
module SPLL(CLK_IN, CLK_OUT);

parameter IN_FREQ = “1”;
parameter CLK_OUT_TO_PIN = “OFF”;
parameter WAKE_ON_LOCK = “OFF”;

input CLK_IN;
output CLK_OUT;

endmodule
```

Simple PLL Parameter Declaration and Instantiation

```
defparam I1.IN_FREQ = “100.0000”,
I1.CLK_OUT_TO_PIN = “OFF”,
I1.WAKE_ON_LOCK = “ON”;

SPLL I1 (.CLK_IN(CLK_IN),.CLK_OUT(CLK_OUT));
```

9.1.2. VHDL, Library Instantiation

```
library lattice;
use lattice.components.all;
```

Simple PLL Architecture and Attribute Declaration

```
architecture behave of simplepll is
component SPLL
generic( IN_FREQ : string;
         CLK_OUT_TO_PIN: string;
         WAKE_ON_LOCK: string);

port( CLK_IN: in std_logic;
    CLK_OUT: out std_logic);

end component;
```

Simple PLL Instantiation

```
begin
I1: SPLL
generic map( IN_FREQ       => “100.0000”,
           CLK_OUT_TO_PIN => “OFF”,
           WAKE_ON_LOCK  => “ON”)
port map(CLK_IN=> CLK_IN,
         CLK_OUT => CLK_OUT);

end behave;
```

9.2. Attribute Examples

9.2.1. Verilog with Synplicity

Persistent

```
/* synthesis PERSISTENT =[ON,OFF] */;
```

Open Drain

```
/* synthesis DONE_OD =[OFF,ON] */;

WAKE_UP

/* synthesis WAKE_UP=[1..25] */;
```

9.2.2. Verilog with Exemplar

Persistent

```
// exemplar attribute [PinName] PERSISTENT [ON,OFF];
```

Open Drain

```
// exemplar attribute DONE_OD [ON,OFF];
```

WAKE_UP

```
// exemplar attribute WAKE_UP=[1..25];
```

9.2.3. VHDL

Persistent

```
ATTRIBUTE PERSISTENT: string;

ATTRIBUTE PERSISTENT IS “[ON, OFF]”;
```

Open Drain

```
ATTRIBUTE DONE_OD : string;
ATTRIBUTE DONE_OD IS “[ON,OFF]”;
```

WAKE_UP

```
ATTRIBUTE WAKE_UP : string;
ATTRIBUTE WAKE_UP:SIGNAL IS “[1..25]”;
```

10. Appendix A. ispXPLD

The following is a summary of minor differences in a Lattice ispXPLD device from the configuration guide for general Lattice ispXP devices.

10.1. Background Read in sysCONFIG and ISP Modes

When a Read command is done in the sysCONFIG or ISP mode for a general ispXP device, the READ is done in the background and the device and I/Os remain functional. For the next generation ispXPLD, the Read command in sysCONFIG and ISP modes is not done in the background. The Read command for SRAM puts the ispXPLD device into configuration mode where the device and I/Os are not functional. The I/Os become tristated.

For more detailed information on ispXPLD read back procedures, please contact Lattice Semiconductor technical support.

10.2. Wake-up Time After Power Up

As Lattice introduces the latest in CPLD technology with the ispXPLD, the best characteristics of a Lattice CPLD should be support, such as “instant on” functional devices, when the power is applied, a Lattice CPLD is ready to go. The next generation ispXPLD device is designed to power up as quickly as possible to maintain the dedicated E^2CMOS power up “instant on”. With the on-board E^2CMOS memory, the SRAM configuration can be configure on power up within t_{WAKEUP} time spec. For ispXPLD devices, t_{WAKEUP} is designed to be as short as possible. t_{WAKEUP} is targeted to 100 µsec of the Vcc's reaching the minimum values. Please check with the ispXPLD device data sheet for the specific t_{WAKEUP} timing.

Technical Support Assistance

Hotline: 1-800-LATTICE (Domestic)
1-408-826-6002 (International)
e-mail: techsupport@latticesemi.com