Lattice Semiconductor Corporation

In-System Programming Usage Guidelines for ispJTAG Devices

February 2002

Introduction

Once a design has been compiled to a JEDEC file and device programming is necessary, the fuse map data must be serially shifted into the device along with the appropriate addresses and commands. Traditionally, programmable logic devices have been programmed on PLD/PROM programmers, so the programmer generates all the programming signals and algorithms. The programmer also generates the external super voltage or high voltage required by non-ISP™ devices (typically 12-14 volts). This super voltage requirement is one of the reasons dedicated programmers are used to program conventional PLDs.

With In-System Programmable (ISP) devices, the ISP programming super voltage is generated within the device from the power supply. Lattice ISP devices use nonvolatile E^2CMOS® technology and require only the device power source for TTL-level programming signals. An integrated state machine controls the sequence of programming operations such as identifying the ISP device, shifting in the appropriate data and commands and controlling internal signals to program and erase the E^2 cells in the device. Programming consists of serially shifting the logic implementation stored in a JEDEC file into the device along with appropriate address and commands, programming the data into the E^2CMOS logic elements, and shifting the data from the logic array out for device programming verification.

The current CPLD industry standard uses the IEEE 1149.1-1990 Boundary Scan Test Access Port (TAP) as the primary interface for in-system programming. Lattice offers more JTAG in-system programmable and testable CPLDs than any other manufacturer. All the ispLSI® 1000EA, 2000VE/VL, 5000V, 8000/V, ispMACH™ 4A, 4000B/C, 5000VG, MACH® 4 and 5, ispGDXV and ispGDX® families are compliant with the IEEE 1149.1 standard while the ispLSI 2000E, ispGAL®22LV10 and ispPAC® families are compatible with it. Some of the original Lattice 5V devices use the proprietary Lattice ISP interface and state machine for programming. For information on using these devices, refer to *Using Proprietary Lattice ISP Devices* on the Lattice CD-ROM or web site at www.latticesemi.com.

The new IEEE 1532 programming standard is fully supported by Lattice. All of Lattice's ispJTAG™ devices are either fully IEEE 1532 compliant or compatible. Based on the industry standard IEEE 1149.1 Boundary Scan Test Access Port, IEEE 1532 compliant and compatible devices can be programmed in the same JTAG chain as other JTAG devices.

Programming Basics

To successfully program devices in-system, there are a few simple requirements that must be met. First, the devices on the board must be correctly connected in an 1149.1 scan chain. This scan chain can be used for either programming or testing the board. To program using the ispVM™ System software, a description of the scan chain needs to be developed. This description is called a chain file and contains basic information about all devices in the chain. For Lattice devices, this includes the device type, the operation to be performed and the JEDEC file, if required by the operation. Additional information in the chain file can include the state of the I/O pins during programming along with security requirements. If non-Lattice devices are present in the chain, the instruction register length or SVF file are required for these devices.

Another requirement for successful programming is thoughtful board design. The signals used in a scan chain (TCK, TMS, TDI and TDO) rarely operate as fast as the data path signals on the board. However, correct board layout methodologies such as buffering for large chains, termination resistors, etc. are required to ensure trouble-free operation. Some Lattice devices have additional pins that can affect boundary scan programming and test if not taken care of properly. For more information on specific board layout recommendations, refer to *In-System Programming Design Guidelines for ispJTAG Devices* on the Lattice CD-ROM or Lattice web site at www.latticesemi.com.

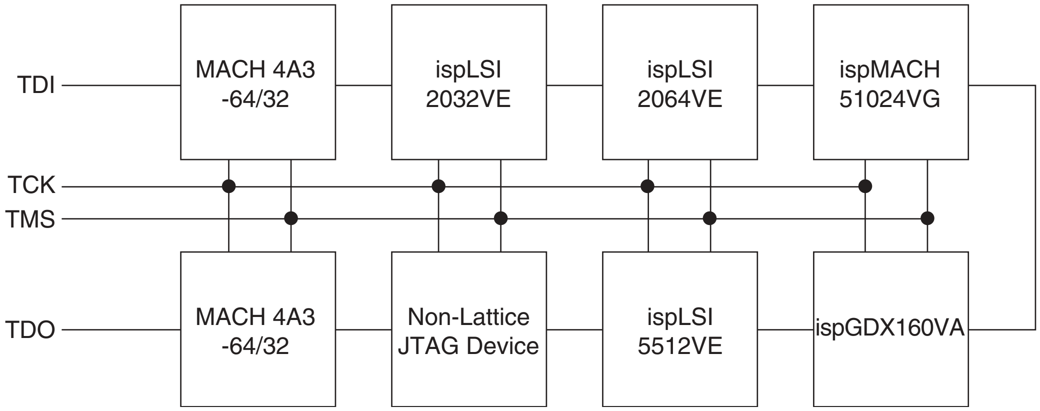
After these requirements have been met, it will be easy to program any number of devices on a board. This programming can be accomplished using a PC or Workstation with a cable attached to the board, a board test system or embedded processor. ispVM System software can be used to facilitate programming in any of these environments.

JTAG Scan Chains

A scan chain can include any IEEE-1149.1 compliant, programmable or non-programmable device. It can also include any programmable devices that are compatible with IEEE-1149.1 but do not have a boundary scan register. This is a decision that should be made based on the test methodology being employed for the board. If the test methodology employed is the traditional bed-of-nails approach used on board test systems, all the devices can be included in the same chain.

All scan chains use the simple four-wire TAP. The TCK and TMS pins are common to all devices included in the chain. TDI and TDO are daisy-chained from one device to the next. The input to the chain is TDI and the output from the chain is TDO. A diagram demonstrating a simple scan chain is shown in Figure 1.

Figure 1. Example JTAG Scan Chain



Programming Algorithm Basics

Programming a CPLD is similar to programming any piece of memory such as an EPROM or FLASH memory. The device can be thought of as an array that is programmed one row at a time. Programming information is provided to the software in the form of a standard JEDEC file that needs to be converted into the row and column data. Before an EEPROM device can be programmed, it first has to be erased. After the device has been erased, the programming data can be loaded and the device programmed. After the device has been programmed, it will be verified by reading the data in the device and comparing it against the original. Figure 2 shows the basic programming flow for the device. It does not include JEDEC file data conversion as it assumes that has already been done.

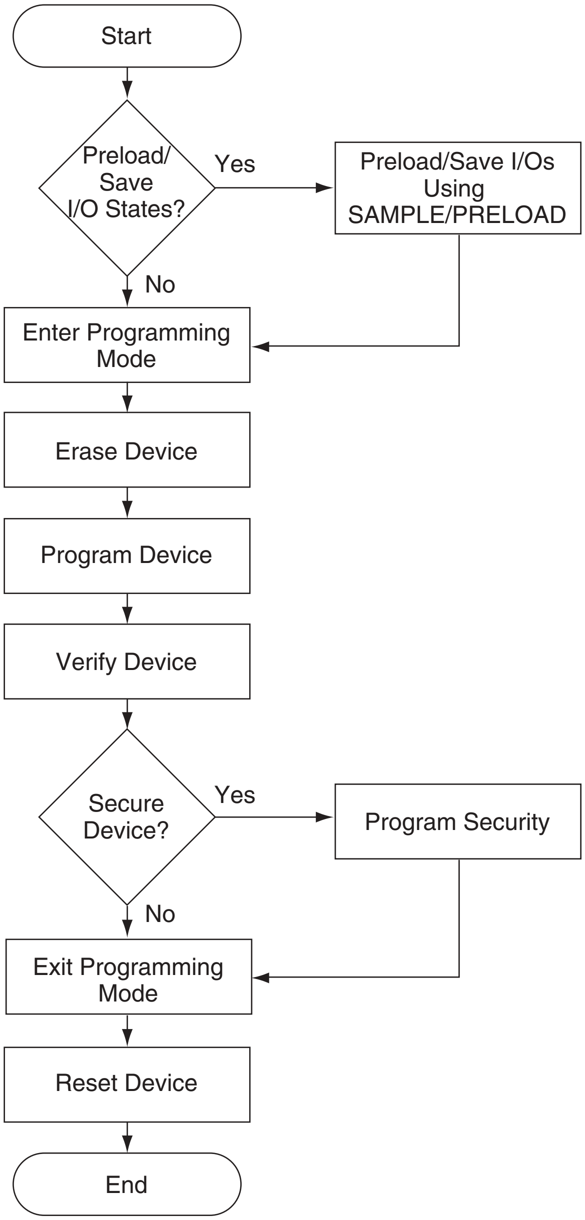
This programming flow will be the same regardless of the programming hardware used. The primary difference between programming on different hardware platforms is the type of data format used.

Programming Times

The time it takes to program a device can often be a determining factor of where in the manufacturing process a device, or group of devices, is programmed. A board test system that costs hundreds of thousands of dollars to purchase and as much as a dollar per minute to operate is an expensive alternative for programming if programming times are long. In most instances, it is more cost effective to purchase several PCs and program devices using these much cheaper systems.

The time it takes to completely program a device is based on the time it takes to first erase the device, then program each row in the device, and then finally to verify the device. The erase time for all devices is the between 100 and 200ms. A single row is programmed in 25 to 80ms, depending on the device. The verify process is the quickest of the required steps in the programming sequence and is mainly dependent on the time required to shift the verify data out of any given device.

Figure 2. Programming Routine Flow



Notes:

1. Although it is not necessary, a reset should always be performed before and after programming a device.
2. If the device will not be programmed in-circuit (i.e. via a cable or using an embedded processor), then it is not necessary to preload or save the I/O states.

To minimize the total programming time of a daisy chain of ISP devices, a programming method called Turbo ispDOWNLOAD® can be used to program all the ISP devices in the chain concurrently. Turbo ispDOWNLOAD allows any number of Lattice ISP devices to be programmed at the same time. When programming a chain concurrently, the chain can be programmed in the time it takes to program only the largest device plus some extra time in order to shift instructions and data for multiple devices. For example, a chain of three devices with programming times of ten, seven and seven seconds can be programmed with Turbo ispDOWNLOAD in a total of about ten seconds (the time it takes to program the largest device). Serially, the programming time would be 24 seconds for all three devices. Turbo ispDOWNLOAD is incorporated into the ispVM software. This valuable feature of Lattice device technology is not available with many other ISP CPLD device technologies.

The minimal programming times will only be seen on board test systems because they are included as a part of the test program and are running at the fastest speed possible. Additionally, there is no translation needed to or from JEDEC-formatted data as this has already been done by ispATE® which is included with the ispVM System software.

USERCODE

User-programmable identification can ease problems associated with document control and device traceability. The ispLSI 1000EA, 2000E, 2000VE, 2000VL, 5000V, 8000/V, MACH 4/A, 4000B/C, 5000VG, ispGDX and ispGDXV families contain a 32-bit register accessible through the optional IEEE 1149.1 USERCODE instruction. This user-programmable ID register is basically a user's "notepad" provided in electrically erasable (E^2) cells on each device. For more information on specific family USERCODE space, refer to the *User Electronic Signature* document on the Lattice CD-ROM or Lattice web site at www.latticesemi.com.

In the course of system development and production, the proliferation of PLD architectures and patterns can be significant. To further complicate the record-keeping process, design changes often occur, especially in the early stages of product development. The task of maintaining which pattern goes into what device for which socket becomes exceedingly difficult. Once a manufacturing flow has been set, it becomes important to "label" each PLD with pertinent manufacturing information, which is beneficial in the event of a customer problem or return. A USERCODE register is incorporated into ISP devices to store such design and manufacturing data as the manufacturer's ID, programming date, programmer make, pattern code, checksum, PCB location, revision number, and/or product flow. This assists users with the complex chore of record maintenance and product flow control. In practice, the user-programmable USERCODE register can be used for any of a number of ID functions.

Within 32 bits available for data storage, users may find it helpful to define specific fields to make better use of the available storage. A field may use only one bit (or all bits), and can store a wide variety of information. The possibilities for these fields are endless, and their definition is completely up to the user.

Even with the device's security feature enabled, the USERCODE register can still be read. With a pattern code stored in the USERCODE register, the user can always identify which pattern has been used in a given device. As a second safety feature, when a device is erased and re-patterned, the USERCODE identification is automatically erased. This prevents any situation in which an old USERCODE might be associated with a new pattern.

It is the user's responsibility to update the USERCODE when reprogramming. It should be noted that the USERCODE information will not be included in the fusemap checksum reading but is included in the verify step when programming or writing.

Loading of the USERCODE instruction makes the USERCODE available to be shifted out in the Shift-DR state of the TAP controller. The USERCODE register can be read while the device is in normal functional operation, allowing the device to be scanned while operating.

Programming Hardware and Software

All ISP programming specifications such as the programming cycle and data retention are guaranteed when programming ISP devices over the designed temperature range as specified in the device data sheet. It is critical that the programming and bulk erase pulse width specifications are met by the programming platform to insure proper in-system programming. The details of device programming are transparent to the user if Lattice ISP programming hardware and software are used.

PC Hardware

Programming is most commonly done on a PC through an ispDOWNLOAD cable attached to the parallel port using the ispVM System software.

ispDOWNLOAD Cable

The ispDOWNLOAD cable is designed to facilitate in-system programming of all Lattice ISP devices on a printed circuit board directly from the parallel port of a PC. The ispVM System software generates programming signals directly from the parallel port of a PC, which then pass through the ispDOWNLOAD cable to the device(s). With this cable and a connector on the board, no additional components are required to program a device. Refer to the ispDOWNLOAD cable data sheet for more detailed specifications and ordering information.

ISP Engineering Kit Model 300

The ISP Engineering Kit Model 300 provides designers with a quick and inexpensive means of evaluating and prototyping new designs using Lattice devices when compared to a stand-alone programmer. This kit is designed for engineering purposes only and is not intended for production use. The kit programs devices from the parallel printer port of a host PC. By connecting a system cable (included) from the host PC to the ISP Engineering Kit with the proper socket adapter, a device can be easily programmed using the ispVM System. Contact a Lattice sales representative for available device and package support.

Workstation Hardware

There is a version of the ispDOWNLOAD cable that allows programming from the Sun Workstation serial port using a command-line utility. Refer to the ispDOWNLOAD cable data sheet for more detailed specifications and ordering information

ispVM System Software

The ispVM System software supports programming of all Lattice ISP devices in a serial daisy chain programming configuration in a PC environment. The software is built around a graphical user interface. Any required JEDEC files are selected by browsing with a built-in file manager. This software supports both serial and concurrent (turbo) programming of all Lattice devices. Any non-Lattice devices that are compliant with IEEE 1149.1 can be bypassed once their instruction register length is defined in the chain description. Using Lattice's revolutionary ispVM software, programmable devices from other vendors can be programmed through the vendor-supplied SVF file.

Programming on a Board Test System

Programming on a board test system is possible by using the ispATE utility (built into the ispVM System software) to generate the necessary programming files needed for the different platforms. The platforms supported include Teradyne, Genrad and Hewlett Packard board test systems. A generic ASCII vector format is generated to help support any ATE not directly supported. Additional information on programming on any of these systems is shown in the ispVM System on-line help.

Programming on JTAG Test Systems

JTAG test systems differ from traditional board test systems in their basic test methodology. These systems use only the four wire JTAG TAP to perform any interconnect and functional tests. A simple language has been developed to interface with the TAP and is used by most major JTAG test system vendors. This language is known as the Serial Vector Format (SVF) and is supported by the ispVM System software. Information on generating a SVF programming file is provided in the ispVM System on-line help.

Embedded Programming

Source code is available for programming devices in an embedded or customized environment. The programming source code is written in ANSI-standard C language which can be easily incorporated into an embedded system or tester software to support programming of ISP devices. This code supports such common operations as Program, Verify, Erase and Secure. After completion of the logic design and creation of a JEDEC file(s), the ispVM System software creates the data files required for in-system programming on customer-specific hardware: PCs, testers or embedded systems.

Technical Support Assistance

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