

## Introduction

In-system programming (ISP™) has often been billed as a direct replacement for configuring a device through a programmer. The idea that devices can simply be placed on a board, connected to a PC through a cable and programmed is an attractive alternative for many newer packages such as the Thin Quad Flat Pack (TQFP) or Ball Grid Array (BGA). When devices are placed on a board, however, care must be taken in the design of that board in the loading of the clock lines, buffering, and termination of signals. This is true for ISP signals as well as the data-path or control signals generated or used by a device. Therefore, it is necessary to follow some general guidelines when designing in-system programmability into a board.

An ideal setup for in-system programming includes buffers at both the parallel port connection of a short cable and on the board to be configured, in addition to termination of all lines which are run in parallel such as TMS and TCK. When there are only a few devices in the chain, the buffer inside the ispDOWNLOAD® cable may be sufficient. The following recommendations will make for a smoother ISP experience.

## Device-specific Connections

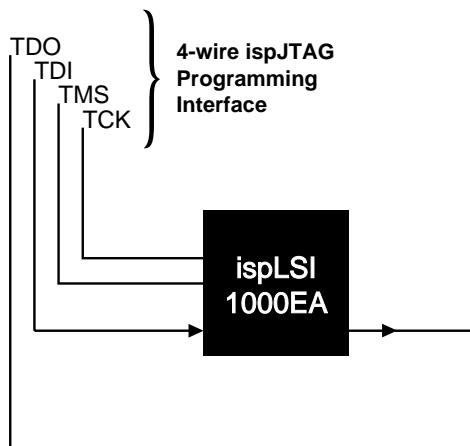
In addition to good PCB layout practices, including the use of decoupling capacitors between  $V_{CC}$  and ground on each ISP device and minimizing trace lengths wherever possible, additional care must be taken to ensure programming interface signal integrity. Many of Lattice ISP devices include some optional pins ( $\overline{\text{ispEN}}$ , BSCAN, ENABLE, EPEN and BSCAN/ $\overline{\text{ispEN}}$ ) that enable the IEEE 1149.1 TAP controller. Some devices support the optional asynchronous reset pin,  $\overline{\text{TRST}}$ . Additionally, the TOE pin is available on some devices to tri-state the functional I/O pins. The following section provides a summary of these optional pins. Unless using some outside control, most of these optional pins are not used on a typical board. A typical wiring configuration is shown for each device family.

For the ispLSI® 2000VE, 2000VL and 2000E devices, a filtering capacitor (.01 $\mu\text{F}$ ) must be provided between the  $\overline{\text{ispEN}}$  signal and ground when the  $\overline{\text{ispEN}}$  or BSCAN device pin is connected to the  $\overline{\text{ispEN}}$  signal on the programming connector. This filtering capacitor must be located as close as possible to the ISP connector on the PCB, in order to filter out any noise during programming. The  $\overline{\text{ispEN}}$  signal is driven low while programming. Without the capacitor, noise can couple into the  $\overline{\text{ispEN}}$  signal during programming and could interrupt the programming sequence.

### ispLSI 1000EA Family

No pins affect TAP controller operation as shown in Figure 1.

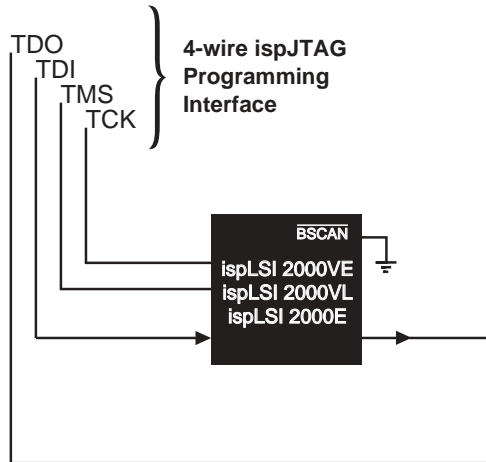
**Figure 1. ispLSI 1000EA Family**



**ispLSI 2000VE, 2000VL and 2000E Families**

For strict boundary scan compliance, the  $\overline{\text{BSCAN}}$  pin must be low. When this pin is high, the TAP controller pins are disabled and the functional dedicated inputs are selected. Since this requires some external control to prevent contention between the functional and boundary scan signals, it is recommended to reserve these pins for boundary scan. Figure 2 shows a recommended connection.

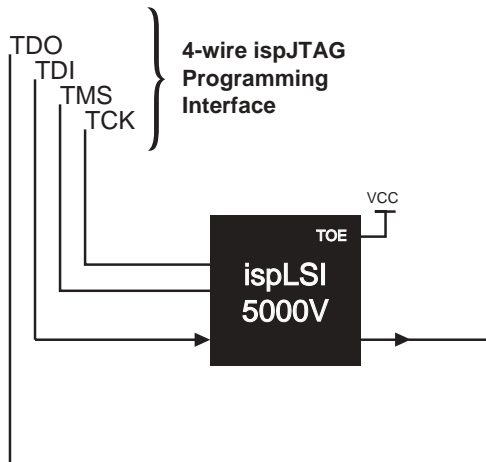
**Figure 2. ispLSI 2000VE, 2000VL and 2000E Families**



**ispLSI 5000V Family**

The TOE pin will disable functional I/O pins when driven low. Unless using boundary scan tester hardware and software that controls TOE, tie it to  $V_{CC}$  to disable this pin. While in programming mode or EXTEST is loaded, the I/O pins are controlled by the boundary scan test registers and TOE has no affect. Figure 3 shows TOE tied to  $V_{CC}$ .

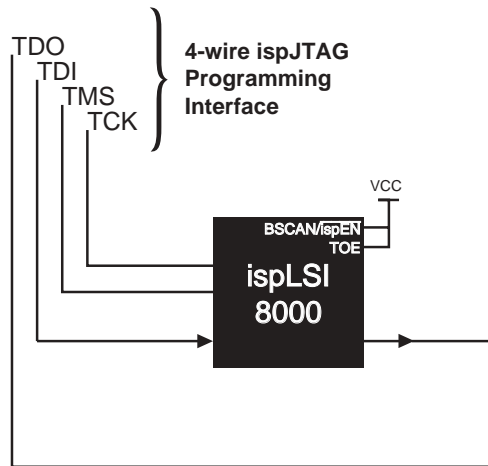
**Figure 3. ispLSI 5000V Family**



**ispLSI 8000 Family**

For strict boundary scan compliance, the  $\text{BSCAN}/\overline{\text{ispEN}}$  pin must be high. When this pin is driven low, the TAP controller pins are disabled, the device enters ISP programming mode and the I/O pins tri-state. The TOE pin will disable the functional I/O pins when driven low. Unless using boundary scan tester hardware and software that controls TOE, tie it to  $V_{CC}$  to disable this pin. While in programming mode or EXTEST is loaded, the I/O pins are controlled by the boundary scan test registers and TOE has no effect. Figure 4 shows  $\text{BSCAN}/\overline{\text{ispEN}}$  and TOE tied to  $V_{CC}$ .

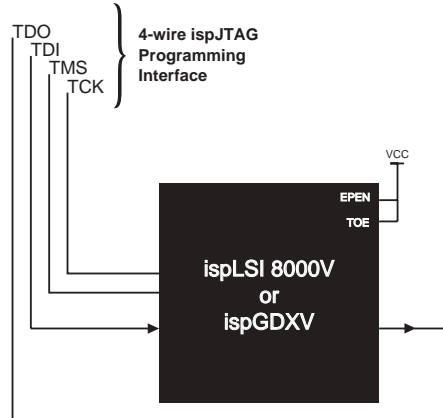
Figure 4. ispLSI 8000 Family



**ispLSI 8000V and ispGDXV Families**

For strict boundary scan compliance, the EPEN pin must be high. When this pin is driven low, the TAP controller pins are disabled. EPEN can be used as a chip select to allow the TAP pins to be connected directly to a bus. The TOE pin will disable the functional I/O pins when driven low. Unless using boundary scan tester hardware and software that controls TOE, tie it to  $V_{CC}$  to disable this pin. While in programming mode or EXTEST is loaded, the I/O pins are controlled by the boundary scan test registers and TOE has no affect. Figure 5 shows EPEN and TOE tied to  $V_{CC}$ .

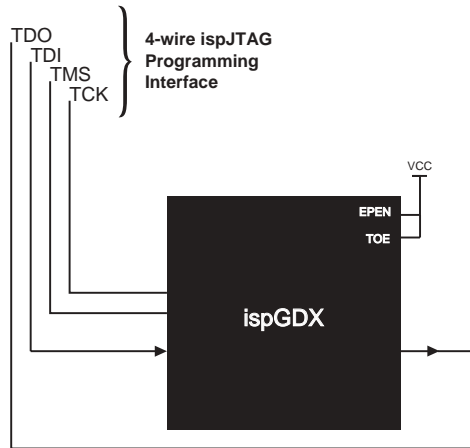
Figure 5. ispLSI 8000V and ispGDXV Families



**ispGDX<sup>®</sup> Family**

For strict boundary scan compliance, the BSCAN/ $\overline{\text{ispEN}}$  pin must be high. When this pin is driven low, the TAP controller pins are disabled, the device enters ISP programming mode and the I/O pins tri-state. The TOE pin will disable the functional I/O pins when driven low. Unless using boundary scan tester hardware and software that controls TOE, tie it to  $V_{CC}$ . Figure 6 shows BSCAN/ $\overline{\text{ispEN}}$  and TOE tied to  $V_{CC}$ .

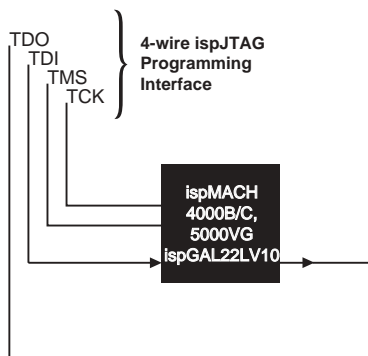
Figure 6. ispGDx Family



**ispMACH 4000B/C, 5000VG, ispGAL<sup>®</sup>22LV10**

No pins affect TAP controller operation as shown in Figure 7.

Figure 7. ispMACH 4000B/C, 5000VG, ispGAL22LV10



**ispMACH<sup>™</sup> 4A Family**

The  $\overline{\text{TRST}}$  and  $\overline{\text{ENABLE}}$  pins are optional JTAG pins and are only found on some of the ispMACH 4A devices, as listed in Table 1. For strict boundary scan compliance, the  $\overline{\text{ENABLE}}$  pin must be low. When the pin is high, the program and verify instructions become inactive. For new designs, it is recommended that the  $\overline{\text{TRST}}$  pin is tied high to keep the TAP controller from being inadvertently reset. If the ispMACH 4A device supports  $\overline{\text{TRST}}$  and the  $\overline{\text{TRST}}$  pin is used, Lattice recommends that a 4.7K $\Omega$  resistor should be connected between the  $\overline{\text{TRST}}$  signal and ground. The resistor will hold the TAP controller in reset until the TRST signal is overdriven by a high signal. Adding the resistor will increase noise immunity at power-up.

Figure 8. ispMACH 4A

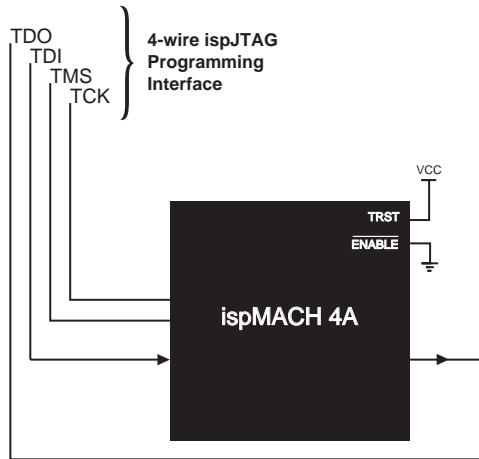


Table 1. ispMACH 4A JTAG Programming Pin Support

Device	Programming Pins Supported
ispMACH 4A (3,5)-32/32	TDI, TDO, TMS, TCK
ispMACH 4A (3,5)-64/32	TDI, TDO, TMS, TCK
ispMACH 4A3-64/64	TDI, TDO, TMS, TCK, $\overline{\text{TRST}}$ , $\overline{\text{ENABLE}}$
ispMACH 4A (3,5)-96/48	TDI, TDO, TMS, TCK
ispMACH 4A (3,5)-128/64	TDI, TDO, TMS, TCK, $\overline{\text{TRST}}$ , $\overline{\text{ENABLE}}$
ispMACH 4A (3,5)-192/96	TDI, TDO, TMS, TCK
ispMACH 4A (3,5)-256/128	TDI, TDO, TMS, TCK, $\overline{\text{TRST}}$ , $\overline{\text{ENABLE}}$
ispMACH 4A3-256/160	TDI, TDO, TMS, TCK, $\overline{\text{TRST}}$ , $\overline{\text{ENABLE}}$
ispMACH 4A3-256/192	TDI, TDO, TMS, TCK
ispMACH 4A3-384/160	TDI, TDO, TMS, TCK
ispMACH 4A3-384/192	TDI, TDO, TMS, TCK
ispMACH 4A3-512/160	TDI, TDO, TMS, TCK
ispMACH 4A3-512/192	TDI, TDO, TMS, TCK
ispMACH 4A3-512/256	TDI, TDO, TMS, TCK

## MACH® Devices

All MACH ISP devices use a standard IEEE-1149.1 test access port (TAP) as the programming interface. The TAP is comprised of four standard pins: TCK, TMS, TDI and TDO. Certain devices also include the optional asynchronous reset pin, TRST, and a program enable pin,  $\overline{\text{ENABLE}}$ . This configuration is found on the MACH355, MACH445, MACH465, MACH4-128/64, MACH4-192/96 and the MACH4-256/128.

In a programming environment, it is only necessary to connect the four standard TAP pins. With the six-pin configuration, while ispVM™ System software supports the use of the  $\overline{\text{TRST}}$  pin and  $\overline{\text{ENABLE}}$  pin, it is not a requirement.

- For new designs, the  $\overline{\text{TRST}}$  pin should be permanently tied to V<sub>CC</sub> and the  $\overline{\text{ENABLE}}$  pin should be tied to ground.
- If the  $\overline{\text{TRST}}$  pin is used, it should be tied to ground through a 4.7KΩ resistor to improve noise immunity.

Making the connections recommended above for the MACH devices will simplify the layout of a board and will eliminate the need for additional buffers for those signals. If the  $\overline{\text{TRST}}$  pin is used, care should be taken to minimize crosstalk with other signals. Additionally, the TRST pin should be tied to ground through a 4.7KΩ resistor. This will

keep the device in reset until that signal is overdriven by the programming hardware. This increases the noise immunity of the device during power-up situations.

## Pin Connections After Programming

After programming and testing has been completed, the question often arises, “What should be done with the ISP port signals?” One of the requirements in the IEEE1149.1 standard for the JTAG port is that both the TMS pin and the TDI pin have internal pull-up resistors. By ensuring that there is a ‘1’ on the TMS pin, inadvertent clocking of TCK will not cause the JTAG state machine to leave its reset state. During power-up, inadvertent clocking on TCK can cause the 1149.1 state machine and instruction register to come up in an undesirable state. To increase noise immunity during power-up, the following recommendation is made:

- A 4.7K $\Omega$  pull-up resistor should be used on the TMS signal and a 4.7K $\Omega$  pull-down resistor on the TCK signal on a board. As the number of devices connected to the TCK/TMS signals increases, the pull-up and pull-down resistor values may need to be adjusted as more internal pull-up resistors are affecting those signals. For a chain with 10 devices, the pull-down resistor value on TCK should be 2K $\Omega$ .

## Buffering

As stated earlier, the ideal scheme for buffering includes buffers at both ends of the cable and buffers for each group of five to eight devices in the programming chain. This case does not cover all situations, however. For example, in a design with only two devices in the chain, the question arises, “Is a buffer needed?” The recommendation for buffering is as follows:

- Buffering is needed for the TCK, TMS, and  $\overline{\text{TRST}}$  lines. It should also be used for the TDI signal into the board and the TDO signal out of the board.
- If there are fewer than five devices in a programming chain, additional buffers are not required, but are recommended. If there are five or more devices, buffering is recommended, as well as a separate buffer for each group of five to eight devices.
- When using a buffer, trace lengths should be balanced to minimize signal skew. A pull-down resistor is required on TCK and a pull-up resistor is required on TMS after the output driver of each buffer.
- The ispDOWNLOAD cable should be used when available. The length of this cable should be no longer than six feet and its length should be minimized when possible.

The TCK, TMS, and optional signals mentioned earlier ( $\overline{\text{TRST}}$ , TOE,  $\overline{\text{ispEN}}$ ,  $\overline{\text{BSCAN}}$ ,  $\overline{\text{ENABLE}}$ , EPEN and BSCAN/ $\overline{\text{ispEN}}$ ) are run in parallel to all 1149.1 and ISP devices on a board. Therefore, these signals will tend to present a larger load to the source driving them. In many cases, this is the ispDOWNLOAD cable connected to a PC parallel port. This buffered cable is no longer than six feet in length. The transmission line effects of both the cable and the traces on the board are the cause for the recommendation of additional buffering on the board itself. The TDI and TDO signals of each device are daisy chained where the TDO of one device will feed the TDI of the next.

The more devices connected to a given signal, the greater the loading on that signal. Therefore, it is necessary to buffer heavily loaded signals and to split the loading of a given signal so that there is a smaller load. This load should also be balanced, both in terms of the number of devices driven by that signal, and the lengths of the traces to each device, so that signal skew does not become an issue.

There are several buffers suitable for use in the programming chain. These include the 74LS244 and the 74LS367. When selecting a buffer, there are two parameters to watch for. The first is the output edge rate. If it is too fast, reflections can become a concern. The second is input voltage requirements. Certain buffers such as the 74HC244 have VIH specifications that are a function of Vcc. In a mixed-signal environment, where there is a 3.3V and a 5V supply on the board, the Vcc of the buffer should be tied to the lower of the two supplies to reduce VIH.

The correct use of buffers on the board can go a long way to either solve existing programming problems or prevent them. It is not the only consideration, however.

---

## Termination of Signals

In any high-speed board or system design, termination of signals is often required to ensure reliable operation. The same is true in an ISP environment. Termination and correct board layout techniques can go a long way to develop a reliable programming setup. Some of the effects of not terminating a signal can include negative overshoot, where a signal will glitch to a negative voltage for a very short period of time (< 2 ns), or double clocking, where a clock signal may have a negative glitch on its rising edge. Both situations can be devastating in an ISP environment. To prevent such possibilities, the following steps should be taken.

- Avoid using buffers with extremely fast edge rates such as the 74F244.
- Terminate the TCK signal either using a balanced termination network on the main trunk of the signal or by using 68Ω resistors in series with each pin the TCK signal is connected to.

## When Buffers Aren't Used

While buffers are recommended for all designs, they may not always be practical in a smaller design where there are only one or two devices in the programming chain. In a situation such as this, there are precautions that can be taken to minimize problems.

If there are noise problems, they can often be cleaned up using a simple RC filter on both the TCK and TMS signals and on the TDI signal into the first device. Additionally, ISP devices may not always have enough drive capability on their TDO pins to either pull up or pull down a signal six feet away at the parallel port. In this situation the following measure can be taken.

- A 4.7KΩ pull-up or pull-down resistor may be necessary on the TDO signal of the last device in the programming chain to reliably switch the signal into the parallel port. This should only be necessary if buffers are not being included as a part of the board design.

## Mixed Voltage Programming Chains

Combining devices with different  $V_{CC}$  voltages into one programming chain can be a challenge. Devices that have 5V or 3.3V  $V_{CC}$  can be combined with no problem because the JTAG pins are all TTL level signals. The 3.3V and 5V devices can be combined in any order.

When 2.5V and 1.8V devices are added onto a board with 3.3V and 5V devices, the voltage levels of the programming pins are not compatible. In order to have any combination of 1.8V, 2.5V, 3.3V or 5V devices, the JTAG pins will need to be referenced to a common voltage, such as 3.3V. Some devices have a  $V_{CCJ}$  pin which can be used to reference all the JTAG pins to a common voltage supplied to  $V_{CCJ}$ . Other devices that do not support the  $V_{CCJ}$  pin will have to have the JTAG pin voltages translated using a translator chip. A translator chip such as the 74LVC07A can take 1.8V inputs and produce 3.3V outputs. When mixing  $V_{CC}$  and programming chains, caution must be used to protect the integrity of the programming chain to reduce programming errors.

## Debugging in the ISP Environment

If all of the above guidelines are followed, programming should be smooth and reliable. Problems that may arise might be the result of improper settings in the ispVM™ System software or improper device configuration.

Errors most commonly occur when the download software is checking the structure of the programming chain by reading the device IDCODEs of all Lattice ISP devices in the chain and checking for a single bit from all other devices. If an error occurs, the following could be wrong and should be considered:

- If the errors returned are either all '1' or all '0,' the following could be wrong:
  - The programming/JTAG connections are incorrect and should be checked.
  - The programming chain has been incorrectly specified. Check both the order of the devices in the chain and the number of instruction register bits in each of the non-Lattice devices.
- If the errors returned are a combination of '1' and '0' and they vary, the TCK and TMS lines may not be either sufficiently terminated or buffered. Refer to the guidelines above.

## **I/O States Before and During Programming**

All ISP devices are shipped from Lattice with a fuse pattern that will put all I/O pins in the high-impedance state prior to programming. This configuration prevents ISP devices from driving unwanted signals to other devices in the system before they can be properly programmed.

During a programming cycle, all Lattice ISP devices default to having their I/O pins tri-stated. In most situations, this should not cause problems. If the state of the I/O needs to be modified during the programming cycle, the ispVM System software can program the I/O cell with a 1, 0 or Z. See the ispVM System on-line help for more information.

## **Conclusion**

The use of the design guidelines and debug techniques presented here will lead to a reliable ISP design and programming flow. ISP offers many advantages over traditional programming techniques, but additional considerations must be taken into account during its implementation, such as proper buffering and termination. This will ensure an effective and productive ISP experience.

## **Technical Support Assistance**

Hotline: 1-800-LATTICE (Domestic)

1-408-826-6002 (International)

e-mail:techsupport@latticesemi.com