



## FPGA Configuration Guidelines

XAPP 090 November 24, 1997 (Version 1.1)

Application Note By Peter Alfke

### Summary

These guidelines describe the configuration process for all members of the XC2000, XC3000, XC4000 and XC5200 FPGA devices and their derivatives. The average user need not understand or remember all these details, but should refer to the debugging hints when problems occur.

The XC2000-, XC3000-, XC4000- and XC5200-family FPGAs share a basic configuration concept, and can be combined in a common configuration bitstream, but there are also small differences among the four families as described below.

Following their initial power-on configuration-memory initialization, these Xilinx FPGAs are configured by a serial configuration bitstream. The byte-parallel configuration modes just activate an internal parallel-to-serial converter, and then use the serial bitstream internally. (Express mode in the XC5200 configures eight bits in parallel, but this mode is not covered in this application note.) The software generates a bitstream that starts with a 40-bit header (48-bit header for XC5200), see Figure 1.

Each device uses a few of the leading “ones” to prepare for configuration, then detects the 0010 pattern and stores the following 24 bits as a length-count value in an internal register. The content of this register is continuously compared against a running counter that increments on every rising CCLK edge. CCLK is either an output (in Master and Asynchronous Peripheral modes) or an input (in Slave Serial and Synchronous Peripheral modes). In all modes, even in Master Serial, it is the externally observable Low-to-High transition on the CCLK pin that causes the internal action. Every CCLK rising edge that occurs while INIT and RESET are High is counted, even during the preamble. Note that XC2000 and XC3000 use quasi-static circuitry which imposes a 5 ms max limit on the CCLK Low time, while XC4000 and XC5200 are completely static and have no max CCLK time limit. This is, of course, only of interest in XC2000 and XC3000 Slave Serial mode, where CCLK is generated by the user.

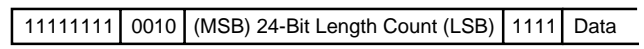
While it is permissible, although not meaningful, to modify the number of leading ones by adding additional ones, or subtracting up to four ones, this would inevitably affect the number of CCLK pulses received by the counter, and thus change the moment when the internal counter is equal to the value stored in the length-count register. **Don't add or delete preamble-leading ones!**

Each device passes the incoming header, including the length-count value, on to the DOUT pin, delayed by half a CCLK period, i.e. the bits are clocked out on a falling CCLK edge. In this way, the header is passed on to all devices that might be connected in a daisy-chain. After the length-count data has been passed on, DOUT goes active High and stays High until the device has been filled with the appropriate number of configuration frames. After that, DOUT again passes all incoming configuration data on to other devices that might be part of the daisy chain.

DOUT is thus the best observation point to see whether the configuration process has started properly.

Immediately following the header, configuration data is received, formatted in a device-specific sequence of frames. Each frame starts with a single “zero” as start bit (XC5200 starts with a byte of seven leading “ones” and a single trailing “zero”), followed by a device-specific number of configuration bits per frame, followed by three “ones” as stop bits (XC2000, XC3000) or, in XC4000 and XC5200, by four bits that are either 0110, or four bits of a running 16-bit CRC error-checking code. The choice is made in the bit-stream generator, where the default is “CRC disabled”. The header is excluded from the CRC calculation.

Each frame is physically shifted into a serial shift register that had been preset to all ones. When the zero start bit hits the far end of this shift register, the data frame is transferred in parallel into the configuration memory, as addressed by the position of an internal token or pointer. The three stop or four error-check bits provide ample time for this transfer, even at a 10 MHz CCLK rate. After this transfer, the shift-in procedure continues with the following frame. Note that there is no counter for the number of bits in the frame nor for the number of frames. The operation is self-synchronized by detecting the presence of a start bit at the far end of the shift register, and by moving the frame pointer.



X5553

Figure 1: 40-Bit Header

Each Xilinx FPGA requires a number of configuration bits that is device-dependent, but independent of the configuration content, and independent of the configuration mode. The number of configuration bits per device ranges from 12,038 for the XC2064 to 1,924,992 for the XC4085XL, approximately 20 bits per available user gate. Exact values are listed in the specific family data sheets.

## Protection Against Data or Format Errors

The serial configuration scheme has proven reliable in thousands of designs and millions of devices, but there have been cases where an erroneous bitstream was loaded accidentally. The original XC2000 and XC3000 devices provide no effective protection against this type of error. If long enough, any random sequence of 0s and 1s will configure such a device. This inevitably takes additional CCLK pulses, more than specified in the length-count value. This means that the CCLK counter already matches the length-count value before the last FPGA in the chain is filled. This comparison is, therefore, ignored, and an additional ~16 million CCLK pulses are required to roll the 24-bit length counter and finish the configuration. Such a configuration will, of course, be wrong and might result in excessive power consumption due to contentions.

XC3000A, XC3100A, XC3000L and XC3100L devices use a simple and effective method to protect against erroneous configuration files or against loss (or gain) of CCLK pulses:

All Xilinx FPGA devices recognize a new frame when its leading zero reaches the end of the shift register. XC2000, XC3000, and XC3100 devices do not check for the presence of valid stop bits, but XC3000A/XC3100A/XC3000L/XC3100L devices always check whether the three bits at the end of the defined frame length are 111. If this check fails,  $\overline{\text{INIT}}$  is pulled Low and the internal configuration is stopped, although a master CCLK keeps running. The user must recognize this state and start a new configuration by applying a  $>6 \mu\text{s}$  Low level on  $\overline{\text{RESET}}$ .

This simple check does not protect against single-bit random errors, but it offers almost 100% protection against erroneous configuration files, defective configuration data sources, synchronization errors between configuration source and FPGA, as well as PC-board defects, such as broken lines or solder bridges.

The XC4000 and XC5200 devices use, optionally, four bits of a running 16-bit cyclic redundancy check code at the end of each frame, combined with additional CRC bits at the end of the bit stream. These error-detecting CRC codes provide excellent protection against errors, even those that do not change the frame structure. When an error is detected,  $\overline{\text{INIT}}$  goes Low and stays Low until the user initiates a reconfiguration. A master device does, however, continue generating CCLK pulses and even incrementing or decrementing the parallel PROM address.

## Daisy-Chain Operation

Multiple FPGAs can be configured by a single concatenated bitstream. The device daisy chain is formed by connecting DOUT to the next device's DIN, and connecting all CCLK pins in parallel. DOUT goes active on a falling clock edge, and DIN accepts data on the subsequent rising clock edge. Each DOUT-to-DIN connection adds one extra bit of delay to the bitstream. Since the header is passed through all devices, they all receive the same header information delayed by one bit per device, but all devices maintain perfect synchronism between their CCLK counters, since all receive the same CCLK.

Xilinx recognizes the need for all devices in a daisy chain to finish configuration and begin user operation simultaneously, as a result of one common CCLK edge. Therefore, all devices in a daisy-chain need a common timing reference. They cannot rely on the start pattern received through the pipelined chain, but must all count the common CCLK pulses exactly the same way. This explains the importance of precise configuration clocking, and the danger of reflections and ringing on the CCLK line.

## Start-Up Procedure

During configuration, all outputs that are not involved in the configuration process are 3-stated, although the crystal oscillator circuit is activated as soon as possible. All internal flip-flops and latches are held reset (set or reset in XC4000), and the DONE output is held Low.

At the end of configuration, these three conditions must change: As shown in detail in Figure 2, the various families offer different options:

**XC2000** has no options; the I/Os go active one CCLK period after length-count match. One CCLK period later, DONE goes active and the global reset is released.

**XC3000** makes the I/Os go active two CCLK periods after length-count match; but DONE and the release of the global reset can each occur either one CCLK period before or after the I/Os go active. The default is "early DONE and late release of the global reset". This makes the outputs go active while the internal logic is still held reset. The other option, "early release of global reset", lets the internal logic be clocked out of its reset state before the outputs go active.

Normally, there is no defined timing relationship between the last configuration events triggered by the rising edge of CCLK, and the subsequent events that are controlled by the system clock. The user must be aware of the potential timing problems of this asynchronous relationship between the two clocks. See the XC4000/XC5200 solution described below.

**XC4000 and XC5200** have more options for the relative timing of I/Os, DONE and GSR, the release of the global set or reset.

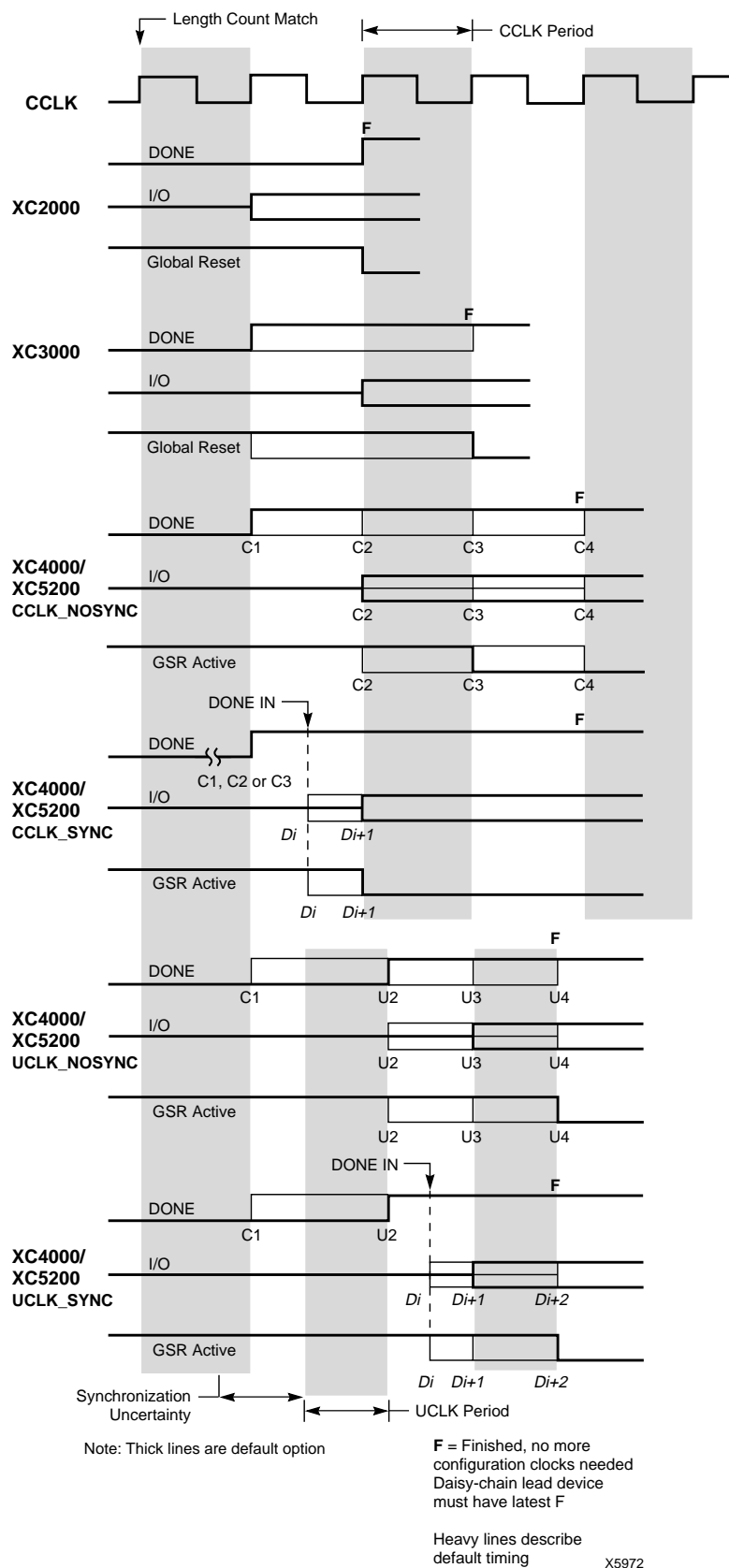


Figure 2: Start-up Timing

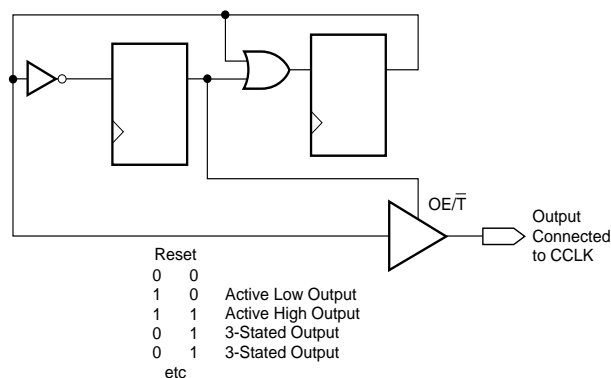
These families can also use DONE as an input to hold off the activation of the I/Os and the release of GSR, until DONE is no longer pulled Low. The change then takes place either immediately upon the release of DONE, or as a result of the next CCLK rising edge. When all DONE pins in a daisy chain are interconnected, this start-up mode guarantees that all devices in the chain go active only when all of them have reached the DONE state, an additional protection against potential configuration errors.

XC4000 and XC5200 can also be configured to employ the system (user) clock instead of CCLK, again either using DONE as an output, or as a bidirectional pin.

The user clock provides a properly synchronized and race-free transition from the end of configuration to the beginning of user operation. The unspecified on-chip delay in the release of GSR (about 100 ns as in XC4013E) requires some caution, however, when using a high clock frequency for configuration.

While devices from different families can be arbitrarily interspersed in a daisy-chain, there is one restriction: the lead device must belong to the highest-numbered family in the chain. If the chain contains XC5200 devices, the lead device cannot be XC4000, XC3000 or XC2000; if the chain contains XC4000 devices, the lead device cannot be XC3000 or XC2000; if the chain contains XC3000, then the lead device cannot be XC2000. The reason is shown in Figure 2. Since all devices in the chain store the same length-count value and generate or receive one common sequence of CCLK pulses, they all recognize length-count match on the same CCLK edge. The master device then generates additional CCLK pulses until it reaches its finish point F. As shown in Figure 2, the different families generate and require different numbers of additional CCLK pulses until they reach F. Not reaching F means that the device has not really finished its configuration process, although DONE may have gone High, the outputs have become active, and the internal reset has been released. For XC4000 and XC5200, not reaching F means that READBACK cannot be initiated, and most boundary scan instructions cannot be used. The limitation in daisy-chain order has been criticized by designers who want to use an inexpensive lead device in Peripheral Mode, and save the more precious XC4000 I/O pins. Here is a solution for that case (Figure 3):

One CLB and one IOB in the lead XC3000 device are used to generate the additional CCLK pulse required by the XC4000 devices. When the lead device releases its internal reset signal, the 2-bit shift register starts responding to its clock input, and it generates an active Low output signal for the duration of one clock period. An external connection between this IOB pin and the CCLK pin thus creates the extra CCLK pulse. This solution requires one CLB, one IOB and pin, and an internal clock source with a frequency of up to 5 MHz. Obviously, the XC3000 lead device must be con-



X5552

Figure 3: Additional CCLK-Pulse Generator

figured with late internal reset, which happens to be the default option.

### Configuration Modes

There are six different configuration modes, hardware-selected by applying logic levels to the three mode inputs, M0, M1, and M2. The six modes are: Master Serial, Master Parallel Up, Master Parallel Down, Synchronous Peripheral (XC4000 and XC5200 only), Asynchronous Peripheral, and Slave Serial. A seventh mode, Express Mode, is only available in XC5200 devices, and is not described here.

In **Master** modes, the FPGA addresses an external PROM or EPROM storage device, and reads data from it. No additional timing or control signals are used.

In **Peripheral** mode, the FPGA accepts byte-wide data (bit-serial in XC2000), and interacts with the source of data, usually a microprocessor, with a Ready/Busy handshake.

In **Slave** mode, the FPGA receives bit-serial data and a clock from an external data and timing source, either from a microprocessor, or from the lead device in an FPGA-daisy chain.

The modes are selected by putting the appropriate logic levels on the three mode inputs, M0, M1, and M2 prior to the beginning of configuration. These three pins can be hardwired to V<sub>CC</sub> or Ground, but they can then never be used as user I/O. It is better to force a mode pin Low with a 3 kΩ pull-down resistor to ground, acting against the 20 to 100 kΩ internal pull-up resistor, and to rely on the built-in pull-up resistor to establish a High level on the M1, M2 mode pins, but use a 50 kΩ external pull-up resistor on M0. This eliminates the restrictions on using mode pins for user logic or readback.

When mode pin levels are driven by external logic, these levels must be established very soon after power-up. Establishing a mode level too late might eliminate the extra master power-on delay that makes a master wait for slave devices to be ready after power-on. Delaying mode levels until the beginning of configuration will obviously cause the

configuration to fail. Note that some CPLD devices have surprisingly long power-up delays. Be very careful when controlling mode levels in any creative way.

## Selecting the Best Configuration Mode

The selection of the most appropriate configuration mode is influenced by many factors, like

- the need for interface simplicity,
- the need for rapid configuration,
- the need for multiple configuration sources,
- the availability of a microprocessor-based configuration driver.

The simplest interface is Master Serial, using only two FPGA pins, CCLK and DIN, and no external timing or control signals.

The fastest configuration mode is Slave Serial or XC4000/XC5200 Synchronous Peripheral. In these modes, the user can supply a well-defined CCLK frequency of up to 10 MHz for 5-Volt devices. Only Express mode can be faster than that. For prototyping and rapid configuration change, the PC can configure the FPGA directly in Slave Serial mode, using the Xilinx-provided Download Cable or XChecker.

Multiple configuration codes are most conveniently stored in a microprocessor memory, using Peripheral mode to configure the FPGA. Peripheral mode also offers the greatest flexibility for field upgrades. New files can be supplied via diskette or modem, and can be downloaded by the microprocessor.

## When Configuration Fails

### General Debugging Hints for all Families

If the DONE output does not go High, there are several things to check.

- Checking all supply and configuration-related pins with an oscilloscope or logic analyzer can reveal wiring errors, bad socket pins, noisy ground, noisy CCLK, a serial configuration PROM's  $V_{PP}$  pin not connected to  $V_{CC}$ ,  $\overline{PWRDWN}$  not pulled High, poor or noisy  $\overline{RESET}$ , missing pull-up resistors on DONE (or  $\overline{INIT}$  in the XC3000), bad levels on mode pins, etc. Check all pins: Any dc voltage between 0.5 V and 3.0 V is a sign of serious trouble.
- Monitor the DOUT pin of the lead device, i.e. the FPGA that is either configured alone, or forms the beginning of a daisy chain. At the start of configuration, you should see the 40 (or 48)-bit header shown in Figure 1. After this sequence, the DOUT pin remains High until the device has received all its data. Then, the device becomes transparent and passes additional data (provided there is a daisy chain) through the DOUT pin to the Slave devices. If you don't see this pattern, you have a gross error somewhere. Check the following

items:

- $\overline{INIT}$  going Low again after configuration start indicates a configuration bitstream or framing error.
- If  $\overline{RESET}$  is used to delay configuration, make sure it has a rise time of <100 ns and that it is glitch-free.
- Ringing on the CCLK line, caused by pc-board reflections, can result in spurious double-clocking and loss of frame synchronization in the FPGA.
- Configuration functions can be disrupted by signal contention between configuration inputs and the FPGA user outputs which become active at the end of configuration. This change is indicated by I/O pins going active and HDC/LDC no longer at their configuration levels. Contention can be avoided by rearranging pin-outs, maintaining additional 3-state control of user-I/O outputs, or matching start-up output levels to the configuration input levels on inputs other than chip-select. As a last resort, it is also possible to use a series resistor (1-10 k $\Omega$ ) to provide isolation between conflicting signal sources that could occur after configuration is complete.
- If an FPGA heats up significantly, this is usually the result of applying the wrong bitstream, e.g. the bitstream for a different device, causing contention. Legitimate bitstreams have been screened by the Design Rule Checker software, and are guaranteed free of inherent contention problems, provided the configuration is loaded into the designated device. The user can obviously still cause contention on internal Longlines and on connections outside the device.
- During reprogramming, user logic must generate a time-out that insures all devices have completed the Clear cycle before any configuration data is sent.
- Removing the FPGA supply voltage while externally powered signals continue to drive input pins, might keep the FPGA  $V_{CC}$  pins at a 0.5-to-2.0 V level, which can leave the FPGA in an invalid state. The FPGA input-protection diodes are there to clamp input-voltage excursions to the two supply connections. When the FPGA supply voltage falls more than 0.5 V below an active input signal, this input signal will supply degenerate  $V_{CC}$  levels. If the input signals are not current-limited, the FPGA inputs can even be damaged by the excessive input current.
- If extraneous CCLK pulses are applied after Clear but before the beginning of the header, they are counted internally, and the internal clock count will then become equal to the stored length-count value before the configuration data is completely loaded. In this case, the DONE output does not become active until the clock counter equals length count a second time. This requires  $2^{24}$  extra clocks, about 20 s at the typical rate of 0.7 MHz, or about 2 seconds at the nominally 8-MHz fastest CCLK rate. Whenever configuration takes several or many seconds, this is due to a mismatch between length count and the number of CCLK pulses

- received.
- XChecker or the XACT Download Cable provide an alternate method of configuration to verify configuration data and to isolate wiring errors, such as interchanged or inverted configuration data or control signals.
- Try a different device. Although the chips are 100% factory-tested, an individual device might have been damaged after the test.

### General Debugging Hints for the XC2000 and XC3000 Families

- An undefined (floating) or active Low PWRDWN during configuration can disturb the operation. A Low level on PWRDWN immediately before the start of configuration causes problems in XC2000, forces XC3000 into Slave mode, but is acceptable in XC3000A and L.
- In the XC2000 and XC3000 families, the configuration-clock input signal drives quasi-static circuitry that does not function correctly with a Low time of more than 5 ms.
- At power-up, make sure  $V_{CC}$  rises in 25 ms or less. If this cannot be guaranteed, hold RESET active on the FPGAs and on the serial PROMs until  $V_{CC}$  has reached 4.5 V.
- A slowly rising or noisy RESET can cause multiple FPGAs to get out of synchronization. Always debounce reset switches.

### General Debugging Hints for the XC4000 and XC5000 Families

- At power-up, make sure  $V_{CC}$  rises in 25 ms or less. If this cannot be guaranteed, hold PROGRAM or INIT active Low on the FPGAs and hold the serial PROMs reset until  $V_{CC}$  has reached 4.5 V.
- The boundary scan input pins are active during configuration, even if boundary scan is not used in the design. Toggling TCK, TMS and TDI during configuration might send the device into EXTEST mode, which interferes with configuration. Keeping at least one of these three inputs continuously High during configuration avoids this problem.

### Additional Mode-Specific Debugging Hints for All Families

#### Master Parallel Up and Down Mode

- Review the general debugging hints.
- Check that the PROM data pins are connected to the FPGA input pins D0-D7. Check that the PROM address pins are connected to the FPGA output pins A0-A15. Verify that all these connections are in the right order. Monitor the FPGA pins, not the socket pins. Make sure the socket is good.
- If the PROM is dedicated to the FPGA, the  $\overline{CS}$  and  $\overline{OE}$  PROM inputs should be driven from the DONE or LDC

FPGA output.

- Verify that the FPGA is sending addresses to the PROM. If it is not, check the FPGA mode pins.

M0 = 0, M1 = 0, M2 = 1 for Master Parallel Up  
M0 = 0, M1 = 1, M2 = 1 for Master Parallel Down

Make sure  $V_{CC}$ ,  $\overline{RESET}$  and  $\overline{PWRDWN}$  are close to  $V_{CC}$  and all ground pins are at 0 V.

- Check that the PROM is receiving addresses and is sending out data. If it is not, check that the PROM is enabled and has  $V_{CC}$  and ground connected, and verify that the PROM is programmed with the correct data.
- Check for contention between the PROM address or data pins and other signals on the board.
- Check that the FPGA is addressing the correct memory segment. In Master Parallel Up mode, the FPGA starts at address 0000 hex and counts up; in Master Parallel Down mode it starts at address FFFF hex (3FFFF hex in XC4000) and counts down. If the PROM requires different addressing, that must be taken care of by external hardware.
- Check for ringing and noise on address and data lines.
- Make sure the data in the PROM is correct. You can check it against the Rawbits file.

#### Master Serial Mode

- Review the general debugging hints.
- Verify that the FPGA is generating a clock signal on its CCLK pin and that this signal is reaching the CLK pin of the XC1700-series Serial-Configuration PROM. If it is not, check the mode pins.

M0 = 0, M1 = 0, M2 = 0 for Master Serial mode

- Verify that the XC1700-series Serial Configuration PROM is sending data. If it is not, check that power and ground are applied to the Serial PROM, and  $V_{PP}$  is connected to  $V_{CC}$ .

#### Do Not Let the $V_{PP}$ Pin Float

**A floating  $V_{PP}$  pin results in temperature-dependent operation, the most notorious cause of unreliable configuration.**

- Check that the DATA pin of the Serial PROM is connected to the DIN pin of the FPGA, and that the PROM is enabled with  $\overline{CE}$  Low and OE active. Note that the OE/RESET pin is programmable for either polarity. Check whether this pin is driven from the INIT output. This is the preferred method of guaranteeing SPROM reset.
- Verify that the PROM is programmed with the correct data.
- At power-up, make sure  $V_{CC}$  rises from 2.0 V to 4.5 V in

less than 25 ms. If it does not, hold the FPGA RESET and the PROM RESET active until  $V_{CC}$  reaches 4.5 V. A typical result of a slow  $V_{CC}$  rise time is that the FPGA sends out CCLK continuously, the  $\overline{CEO}$  pin on the PROM(s) goes Low, but the DONE pin never goes High.

- If you abort configuration by asserting XC3000  $\overline{RESET}$  or by pulling XC4000/XC5000 PROGRAM Low, you must also reset the serial PROM by asserting its RESET. This occurs automatically if the SROM is reset from INIT.

### Asynchronous Peripheral Mode

- Review the general debugging hints.
- Check the mode pin levels.

$M0 = 1, M1 = 0, M2 = 1$  for Peripheral mode

- Use an external 1 kilohm resistor from READY/BUSY pin to ground. On power-up, before the FPGA has interrogated the mode lines, this prevents the pin from being pulled High by its internal pull-up, which would give an early erroneous READY signal.
- Verify that the FPGA is receiving data at its input pin(s) and that it is receiving valid Write-Strobe and Chip-Select signals. If not, check the device driving the FPGA. Make sure that these signals meet the timing requirements listed in the product family documentation. XC3000 Family: Check that the minimum Write-Strobe active time ( $T_{CA} \text{ min} = 100 \text{ ns}$ ) is met and observe the RDY/BUSY signal. XC2000 Family: Be sure maximum and minimum Write-Strobe active times ( $T_{CA} \text{ max} = 5.0 \text{ ms}$ ,  $\text{min} = 0.25 \text{ ms}$ ) are met.
- Make sure that the FPGA is ready to receive data. XC3000 Family: On power up, make sure that the INIT pin has gone High, or wait at least 34 ms before you begin sending data to the FPGA. Make sure that the RDY/BUSY signal is High before sending each data byte. XC2000 Family: On power up, make sure that the FPGA has had time to "wake up," at least 34 ms, before sending it data.
- Check for contention between the Chip Select and Write Strobe signals and monitor the levels on those pins after configuration. It is safest to use the Chip Select pins only as inputs after configuration. Avoid contention if they are used as outputs. With XC2000 family devices, the I/Os become active before the FPGA receives its final data bits and clocks, and also before the DONE pin goes High. In other families, this relative timing is programmable. If the user function for any of the Chip Selects or the Write Strobe become outputs after configuration, they might contend and, in effect, de-select the FPGA so that it never receives its final data bits. Beware of contention!
- Check for contention between the FPGA pins and other

signals on the board. Except in XC2000, data is received as eight bits in parallel. Make sure bit 0 is connected to the D0 pin, bit 1 to D1 pin, etc. (In XC2000 family, data is received serially. If a PROM file is used as a data source, check that data is properly serialized LSB first. Data must be LSB first, although length count is MSB first. This is not intuitively obvious.)

### Slave Serial Mode

- Review the general debugging hints.
- Check the mode pin levels.

$M0 = 1, M1 = 1, M2 = 1$  for Slave Serial mode

- See schematics in the data sheet for the FPGA family.
- Make sure  $V_{cc}$ ,  $\overline{RESET}$ , and  $\overline{PWRDWN}$  are at 5 V, and ground pins are at 0 V.
- Verify that the FPGA is receiving data on DIN and that it is receiving a valid clock signal on CCLK. Check the device sending the data. Check the device sending the clock signal, and make sure the clock meets the timing requirements specified in the product family documentation. Don't violate the XC3000 and XC2000 CCLK Low time specification of 5.0  $\mu\text{s}$ . A CCLK generated by a Master FPGA automatically meets the timing requirements.
- Make sure the FPGA is ready to receive data.
  - XC3000 Family:** On power up, make sure the  $\overline{INIT}$  pin is High or wait at least 34 ms before you begin sending data to the FPGA.
  - XC2000 Family:** On power up, make sure that the FPGA has had time to "wake up" at least 34 ms, before sending it data.
- At power up, make sure  $V_{CC}$  rises from 2.0 V to 4.5 V in less than 25 ms. If it does not, hold  $\overline{RESET}$  Low until the  $V_{CC}$  pins reach 4.5 V.

### Daisy Chain Debugging Hints

- The key to debugging daisy-chain configurations is to isolate the problem and attempt to configure a single FPGA. Remove all but the first device from the board and configure it. Then insert the second device and configure both. Repeat as you add one device at a time until they all configure.
- The first device in the chain can be in any of the configuration modes. Debug it first, using the hints provided for the appropriate mode.
- All devices after the first one are in Slave Serial mode, so refer to the Slave Serial mode debugging hints above to solve any problems with Slave device.
- Monitor the DOUT pin of each device in the chain and verify that the 40-bit header (48-bit with XC5200 as the lead device) appears at the beginning of configuration, staggered by one CCLK period per device.
- If the Master device in the chain is an XC2000-family device and the Slaves are XC3000-family, make sure

the XC3000-family devices are configured with early DONE.

### Potential Length-Count Problem in Parallel or Peripheral Modes

It is highly desirable that the complete change from configuration to user operation occur as the result of one single byte-wide input. The activation of outputs and DONE, the de-activation of the global reset (set/reset in XC4000), and the progression to the “finished” state F (see Figure 2) should all occur as a result of one common byte input. Under normal circumstances, the software achieves this by manipulating the length-count value appropriately, taking into account the additional bits between devices, and adjusting for the fact that byte-wide interfaces always leave the last bit sitting in the P-S converter, shifting it out at the beginning of the next byte. These complexities, combined with the many possible daisy-chain arrangements have occasionally led to problems, where the device outputs go active before the last required byte had been received. This has sometimes lead to contention on the address outputs or data inputs and might prevent the device from going DONE, or reaching the real end of its configuration sequence. Not reaching this “finished” state limits the use of readback and boundary scan. A new option solves this problem:

The default option is “Length-Count aligned” which adjusts the length-count value such that length-count match occurs during the first bit in the last configuration byte. This assures sufficient CCLK pulses to complete any selected type of start-up sequence. The other option is “DONE-aligned”, which adjusts the length count value to make DONE go active at the end of a configuration data byte, which can cause problems in Peripheral mode.

**Only Peripheral modes seem to be sensitive to the difference between these two options.**

## Miscellaneous Notes

**CCLK** is the most important configuration signal. Once the  $\overline{\text{INIT}}$  output is High, each device counts every Low-to-High transition of this configuration clock. In all modes except Slave Serial and Synchronous Peripheral, CCLK is a very fast output that cannot be made slew-rate limited. (it is now slew-rate limited in the newest XC4000X and XC5200 devices). When distributing this clock, the user should pay special attention to glitches, overshoots, and undershoots. In severe cases, a 33  $\Omega$  resistor in series with the CCLK output might improve the signal integrity. In other cases, it might be better to provide a pull-up resistor at the far end of the CCLK net. Since the clock net has a transmission-line characteristic impedance of always less than 100  $\Omega$ , the limited output drive capability of the CCLK output precludes proper parallel termination.

**DOUT** is an excellent observation point, since every device must output the preamble on this pin, irrespective of the selected configuration mode, and irrespective of the position in, or the existence of, a daisy chain.

$\overline{\text{INIT}}$  of all devices in a daisy chain should be interconnected to prevent the configuration from starting before all devices are ready. A 10 k $\Omega$  pull-up resistor is recommended. The parallel  $\overline{\text{INIT}}$  of the daisy-chained devices must be connected to the  $\overline{\text{INIT}}$  of the lead XC4000/XC5200 device, or to the  $\overline{\text{RESET}}$  input of the lead XC3000 device. This is especially important for re-configuration, where the master does not have a four-times longer wait period.

The **DONE** output indicates the end of the configuration process. In XC2000 and XC3000 systems, it makes sense to ground DONE permanently. The  $\overline{\text{RESET}}$  input then becomes the reconfiguration input, and cannot be used as the dedicated asynchronous user  $\overline{\text{RESET}}$  input. LDC can be used to indicate end of configuration.

**PWRDWN** (on XC2000 and XC3000 devices) must be High before and during the configuration process.

Don't let  $\overline{\text{PWRDWN}}$  float!