XAPP079 (v1.1) July 27, 2000

Configuring Xilinx FPGAs Using an XC9500 CPLD and Parallel PROM

Authors: Chris Dunlap, Tom Fischaber

Summary

All Xilinx FPGA families can be configured through a serial interface. This application note describes a simple, low cost design to configure any Xilinx FPGA in a serial configuration mode using a Xilinx XC9500 CPLD and any parallel PROM.

Introduction

As density increases in FPGA families, current serial PROMs cannot handle the programming patterns required to configure high-density FPGAs. Larger PROMs have been made available but many require addressing schemes that make them unsuitable to interface directly to an FPGA. This application note describes how a small XC9500 CPLD can be used in conjunction with a byte-wide PROM to program high-density FPGA devices.

Configuration Steps

Reset the Configuration Logic

If the device is to be configured upon power-up, or after the recycling of the power source, the configuration logic will be automatically cleared during this initialization time. Therefore, it is essential that the system power supply rise quickly and monotonically. If the device is to be reconfigured without recycling the power, then PROGRAM must be pulsed Low for a minimum of 300 ns to reset the configuration logic.

Time-Out Start of Configuration

After the release of the PROGRAM input, or the powering up of the FPGA, the INIT output remains Low while the internal configuration memory is cleared. Once the INIT pin transitions High, the device is ready to be configured. However, there are some additional considerations that should be noted, depending on the device that is being configured.

XC4000/Spartan® Family

In the XC4000 and Spartan family devices, following the transition High of INIT there is an additional time-out period required before the device is ready to accept configuration data (Note: this does not apply to the Spartan-II devices, see Spartan-II Family for details on this device). In a Master Serial mode, the FPGA takes this into account as it simply holds off the start of the Configuration Clock (CCLK) until the time-out requirement has been met. In a Slave Serial mode, special care must be taken to delay the start of CCLK until this time-out period has expired. Refer to the Xilinx device specific specification sheet for information, and see "VSPROM Design Considerations" on page 3 for design recommendations.

Spartan-II Family

Spartan-II does not require an additional time-out period following the transition High of the INIT signal. Thus, once the INIT signal goes High, the device is immediately ready to accept configuration data.

Virtex™ Family

Virtex does not require an additional time-out period following the transition High of the INIT signal. Thus, once the INIT signal goes High, the device is immediately ready to accept configuration data.

Interface Design

To configure a Xilinx FPGA in a serial configuration from a parallel PROM, a small interface is needed to generate the data addresses, perform the parallel-to-serial data conversion, and monitor the FPGA configuration signals. While this interface can be performed in a variety of technologies, a Xilinx XC9500 CPLD is a simple and cost effective implementation, as shown in Figure 1.

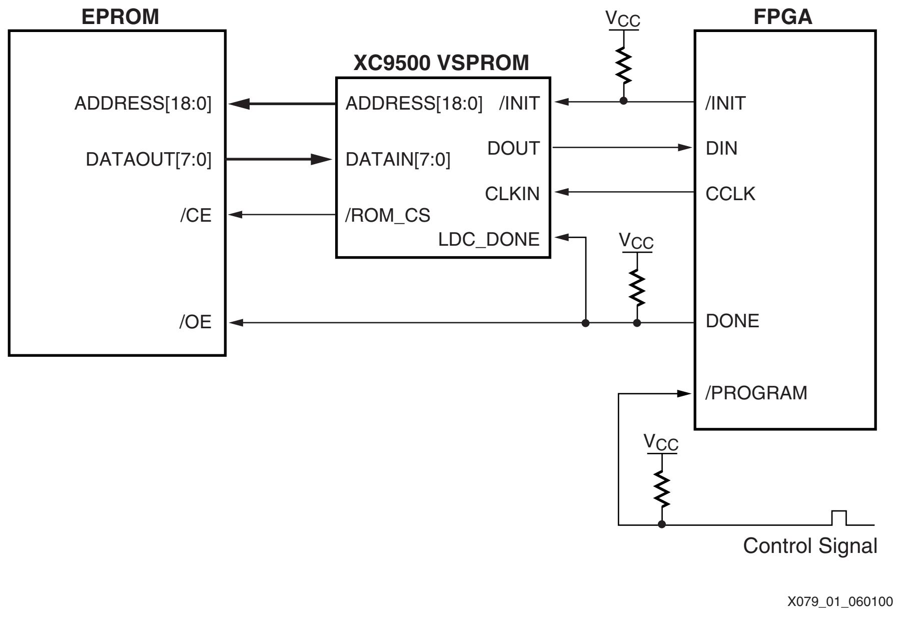

Figure 1: **XC9500 VSPROM Design Interface to FPGA**

Voltage Supplies and Considerations

Power supply consideration is vital in the system to ensure that all devices are properly activated. If a slow rising or non-monotonic power supply is used, problems can arise during power-up that will prevent the devices from configuring correctly. Required power supply rise times can be found in the appropriate family data sheet. Refer to the Xilinx web site: http://www.xilinx.com/partinfo/databook.htm.

All XC4000 and Spartan devices are 5V compliant, and can be directly interfaced to either a 5V or 3.3V environment. The Virtex and Spartan-II devices, however, require some care in interfacing to other logic. These devices are 5V compliant, and thus can be interfaced directly to either 5V or 3.3V environments. However, this assumes that the default I/O standard of LVTTL is used in conjunction with a 3.3V V_{CCO}. If this is not the case, refer to the specific data sheet for voltage compatibility on the Xilinx web site: http://www.xilinx.com/partinfo/databook.htm.

Virtex-E Voltage Interface

Virtex-E devices are not directly 5V tolerant, and thus special consideration must be taken when interfacing a 5V device to a Virtex-E device. For specific 5V tolerance, refer to solution record 7880 at http://support.xilinx.com/techdocs/7880.htm If the VSPROM is to be used with a Virtex-E device, consideration of a XC9500XL device might be the best solution, as this device can directly interface to the Virtex-E device.

The VSPROM Design

The Virtual Serial PROM (VSPROM) design is a small interface to a parallel PROM which allows a Xilinx FPGA to be configured in a serial manner. The design performs a parallel to serial conversion of each byte of data read from the PROM before incrementing to the next address location. The basic outline of the design is described in Figure 2.

The VSPROM design monitors the INIT and DONE lines as it parses data. During configuration, if INIT is pulled Low by the FPGA, indicating a configuration error, the VSPROM design will be reset and wait for the user to reset the FPGA so the device can be reconfigured.

Once DONE goes High on the FPGA, indicating a successful configuration, the VSPROM isolates itself and the PROM from the FPGA by deasserting ROM_CS to disable the PROM and placing DOUT in a High-Z state.

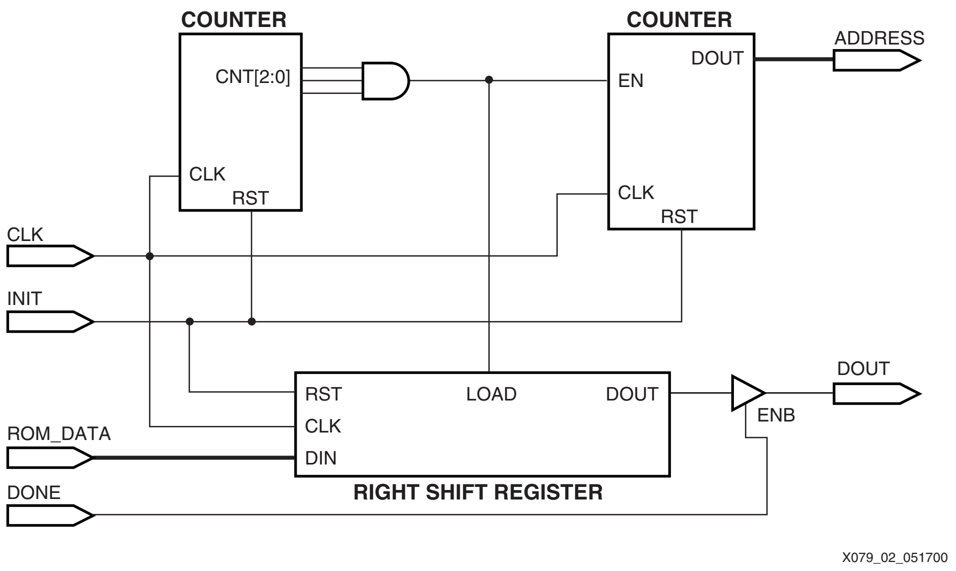

Figure 2: **VSPROM Design Layout**

VSPROM Design Files

The VSPROM design was built in a XC9536PC44-10 and tested with a XCV600BG432-6 and an 8-Mbit PROM (AT27C080). The VHDL and Verilog source code for the design can be downloaded from the Xilinx website at ftp://ftp.xilinx.com/pub/applications/xapp/xapp079.zip.

VSPROM Design Considerations

Master Serial Configuration

For a Master Serial configuration, the Xilinx FPGA will provide the CCLK for configuration. The CCLK should be connected directly to the VSPROM clock (CLK_IN). The maximum configuration rate supported by the Virtex and Spartan-II devices is 60 MHz, and the XC4000/ Spartan families can support up to an 8 MHz configuration rate. However, the access time of the PROM (T_{ACC}) plus the setup time for the Master Serial inputs (T_{DSCK}) must also be taken into account to determine the maximum frequency.

PROM Frequency in Master Serial mode =

$$\frac{1}{T_{ACC} + T_{DSCK}}$$

The T_{DSCK} is 5.0 ns for Virtex, and 20.0 ns for the XC4000/Spartan families. A typical T_{ACC} for a PROM (e.g., AT27C080) is 100 ns. Using these values, the maximum frequency of the oscillator is 9.5 MHz for the Virtex device, and 8.3 MHz for the XC4000 / Spartan devices. In this case, the Virtex device is limited by the PROM configuration, while the XC4000/Spartan device is limited by the internal CCLK rate. For faster configuration of Virtex devices, a faster PROM can be used—up to the maximum speed of either the Virtex device or the VSPROM CPLD design.

If configuring in Master Serial mode, the VSPROM design can be applied without any special considerations since the FPGA will hold off the CCLKs, if required, until the FPGA is ready to accept the configuration data.

Slave Serial Configuration

Slave Serial configuration is the fastest serial configuration, as an oscillator is used to provide the configuration clock. The clock from the oscillator must be connected to the FPGA Configuration Clock (CCLK) and the VSPROM clock (CLK_IN). The maximum configuration rate for Virtex is 66 MHz, and XC4000/Spartan families can support up to a 10 MHz configuration rate. However, the access time of the PROM (T_{ACC}) plus the setup time for the Slave Serial inputs (T_{DCC}) must also be taken into account to determine the maximum frequency.

PROM frequency in Slave Serial mode =

$$\frac{1}{T_{ACC} + T_{DCC}}$$

Refer to the individual device data sheet for specific values, and see Master Serial Configuration for an example calculation.

When configuring an XC4000 or Spartan series FPGA in a Slave Serial mode, special considerations must also be taken in regards to the start-up sequence (Note: This does not apply to the Spartan-II family). As described in the "Time-Out Start of Configuration" on page 1, once INIT is High, an additional time-out period is required before the device can accept configuration data. Thus, the interface or VSPROM design must be modified to account for this. The interface can be modified by simply holding off the clock output from the oscillator until the required time-out period has expired. The VSPROM design can be modified to wait an additional time after INIT is High before enabling the outputs and commencing with the configuration data, or adding additional dummy bits to the bitstream. See "Appendix A – XC4000 and Spartan Slave Serial Configuration Considerations" on page 8 for more details.

If a Virtex or Spartan-II device is targeted, no special considerations are required, as once INIT transitions High, the FPGA is immediately ready to accept configuration data.

PROM Address Requirements

The size and number of parallel PROMs needed will be determined by the number of configuration bits required to configure the Xilinx FPGA(s). Proportional to the size of the PROMs are the number of required address lines. By default, the VSPROM design uses 19 address lines, corresponding to a 4 Mbit PROM. However, if more memory for configuration storage is required, the VSPROM design can easily be adjusted by modifying the appropriate statement in the source code, GENERIC in VHDL or `define in Verilog. See Table 1 for details. Note: If more than 19 address lines are required for configuration, an XC9572 device should be used to allow a successful fit of the design.

Table 1: **Required Address Lines for Corresponding Configuration Bits**

EPROM	Configuration Bits	Address Lines Needed
1M EPROM	1,048,576	17
2M EPROM	2,097,152	18
4M EPROM	4,194,304	19
8M EPROM	8,388,608	20

The PROGRAM Line of the FPGA(s)

If an error occurs during configuration of a Xilinx FPGA, INIT is pulled Low and the configuration sequence will stop. In this case, the VSPROM design will monitor INIT, and when INIT is pulled Low, the VSPROM design will be reset and wait for the FPGA to be reset before issuing a reconfiguration. There are many possible interface options that the user can consider. A simple modification of the VSPROM design would allow the VSPROM to pulse PROGRAM Low if INIT was pulled Low during configuration, forcing the FPGA to clear its configuration memory, and then issue a retry. However, more complex alternatives are also available, such as interfacing the INIT line to a microcontroller, and performing a board level reset if INIT is pulled Low during configuration. In either case, this decision is left up to the user, as the VSPROM design can easily be modified to accommodate a wide variety of solutions.

Interfacing Multiple PROMS

The largest FPGAs can require configuration bits not achievable by current PROMs. However, the configurable nature of the VSPROM design can easily accommodate multiple PROMs. To daisy chain PROMs, as shown in Figure 3, simply drive the chip enable (/CE) of the PROM with an address decode of the most significant bits of the address bus.

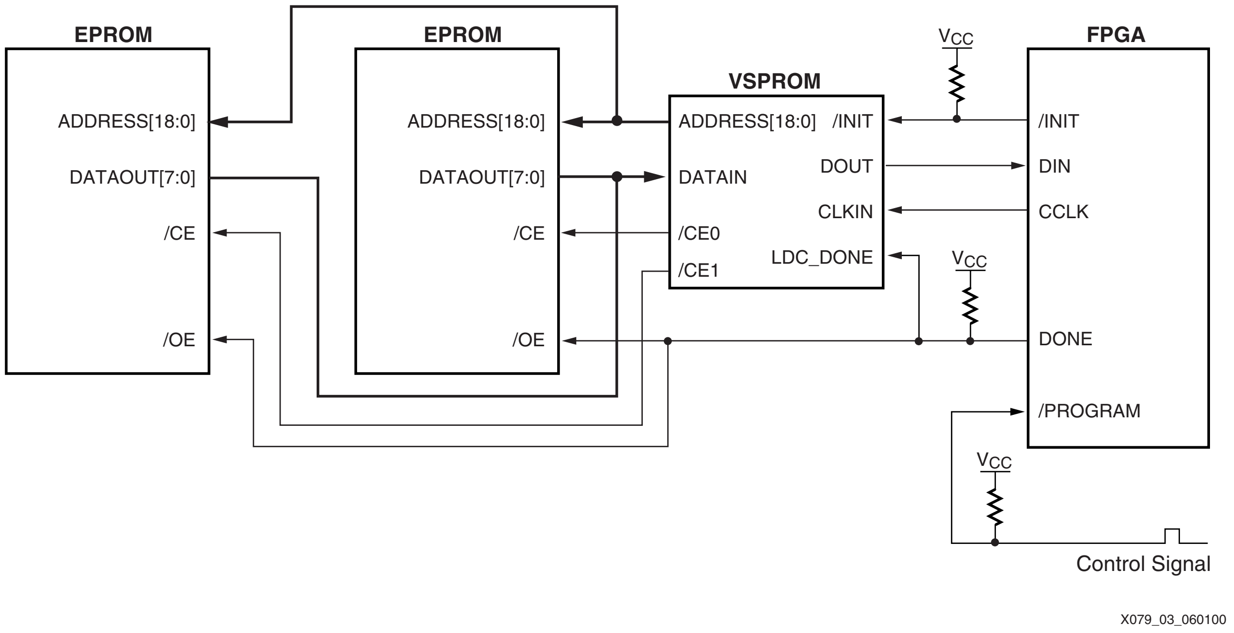

Figure 3: **Connecting Multiple PROMs in a Daisy Chain**

For the example shown in Figure 3, the common address bus is now 20 bits wide and the MSB ADDRESS(19) is used to select between the two PROMs, as shown in Table 2.

Table 2: **Address Decode for Two PROM Selection**

Extra Address Line (A19)	/CE0	/CE1
0	0	1
1	1	0

The /CE outputs are simply a binary decode of an additional MSB of the address bus. To add more PROMs simply increase the size of the address bus and binary decoder. See Figure 4.

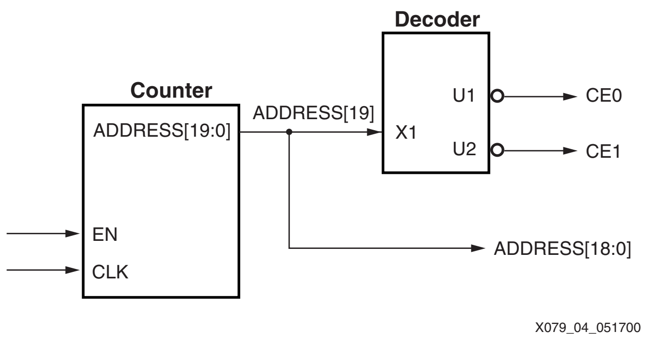

Figure 4: **Address Decode for Cascading PROMs**

Daisy Chaining Multiple FPGAs

Multiple FPGAs can be daisy chained directly, with no special considerations or modifications required of the VSPROM design. For specific recommendations regarding the details of FPGA daisy chaining, refer to the individual target device data sheet. However, in general, CCLK, INIT, DONE and PROG should all be connected in parallel. The DOUT signal of the first device connects to the DIN of the second device. The DOUT of the second FPGA would then connect to the DIN of the third device, and so forth. See Figure 5. All of the bitstreams should be concatenated using either the PROM File Formatter, or promgen.

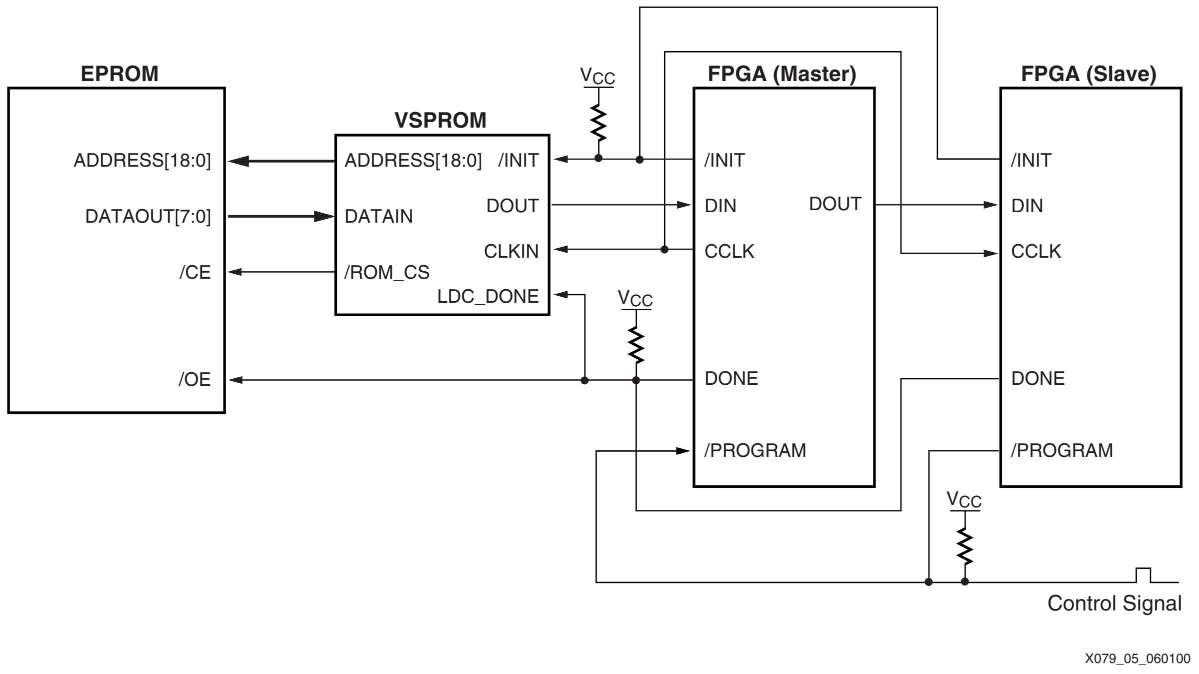

Figure 5: **Connecting Multiple FPGAs in a Daisy Chain**

FPGA Configuration Debugging and Troubleshooting Techniques

For advanced configuration troubleshooting information, refer to the Configuration Problem Solver available at http://support.xilinx.com.

Board Level Considerations

Verify that the CCLK and DIN signals on the FPGA are clean and noise-free, and that the board is adequately decoupled. In general, consider using a 0.1 µF and a 0.01 µF capacitor at every V_{CC} and GND pair.

When powering up a setup that uses the VSPROM, be sure that V_{CC} rises quickly and monotonically.

To the extent possible, allow access to the JTAG pins of the VSPROM. Leaving access to the XC9500s JTAG pins ensures that modifications to the VSPROM can be performed at any time.

On the CCLK line, it may be necessary to add a line driver or termination, if there are problems with clock distribution or reflections. For board level debugging of signal integrity issues, a board level simulator might be considered.

Data Stream Format

Verify that the configuration data is being shifted into the FPGA in the proper order. For example, looking at the 4000 Series Data Stream format in the product specification, it shows that the bitstream will consist of fill bytes of FFh followed by a preamble of 0010b. Refer to the table below for a visual depiction of the data being shifted into the FPGA, CLKx being shifted in first. Note that Clkx+4 represents the start of the preamble for the 4000 family.

Clkx+7	Clkx+6	Clkx+5	Clkx+4	Clkx+3	Clkx+2	Clkx+1	Clkx
0	1	0	0	1	1	1	1

Appendix A – XC4000 and Spartan Slave Serial Configuration Considerations

If an XC4000 or Spartan device is to be configured in a Slave Serial mode, special consideration must be taken to ensure that the FPGA is ready to accept data before the VSPROM design begins attempting to configure the FPGA. Specifically, there is a CCLK delay that must be integrated into the design for the VSPROM to successfully configure. Refer to the device data sheet, Configuration Switching Characteristics, for the specific recommendations.

As an example, the CCLK delay (T_{ICCK}), as shown in the device data sheet, is listed as 4 µs. The maximum configuration rate is 10 MHz. Therefore, the maximum number of clock cycles that could be seen between INIT going High and the device ready to accept data is 40 clock cycles (4 µs/100 ns).

Following are two possible solutions—either modify the VSPROM design to delay an additional 40 clock cycles, or add an additional 40 dummy bits to the configuration bitstream.

Modify the VSPROM design

The VSPROM design can easily be modified to wait an additional time-out before commencing configuration. In this case, simply create a counter that begins when INIT goes High. Once the desired count length is met (in this example, count = 40), then initialize the outputs and begin the configuration sequence.

Modify the Bitstream

The configuration bitstream can also be modified to add additional dummy bits before the Preamble (see the Xilinx product specification for specific details on the bitstream format). There is an internal configuration counter that begins counting CCLKs once the INIT line goes High. When the Length Count is equal to the number of CCLKs seen by the FPGA, the start-up sequence begins. Thus, if additional bits are added in front of the bitstream, the Length Count will also need to be modified to account for this. Since the Xilinx bitstream is in binary format, a HEX editor will be required to modify this. It is also possible to create an ASCII version of the bit file (.rbt file), and this file can be modified directly with any text editor.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
07/27/00	1.1	Updated and re-released.