XILINX®

Configuring Virtex FPGAs from Parallel EPROMs with a CPLD

XAPP 137 March 1, 1999 (Version 1.0)

Application Note by Carl Carmichael

Summary

Previous generations of Xilinx FPGAs supported a Master Parallel Configuration Mode which allowed the FPGA to configure itself directly from a parallel (byte wide) PROM. The Virtex family of Xilinx FPGAs does not utilize a Master Parallel mode. This appnote describes a simple interface design to configure a Virtex device from a parallel EPROM using the SelectMAP configuration mode.

Xilinx Family

Virtex & XC9500

Introduction

Many FPGA users prefer to store configuration data on parallel EPROMs because they are available in greater storage capacities than serial PROMs. A parallel PROM stores data as an addressed byte which is accessed by an address bus.

Configuring a Virtex FPGA through the SelectMAP mode is 8 times faster than bit-serial mode. The SelectMAP configuration mode utilizes an 8 bit configuration bus and 7 control signals for synchronization and handshaking of data.

Combining larger memory capacities with an 8 bit interface results in a faster configuration scheme that requires fewer storage elements thus reducing board utilization and necessary time for system initialization. This is made possible by the addition of a small interface design to provide the address bus for the parallel PROM and the necessary control signals for SelectMAP configuration.

SelectMAP in Virtex

SelectMAP uses a byte wide bidirectional access port for reading and writing the configuration data of a Virtex FPGA. As a configuration mode it is very similar to the Express mode used by XC4000XLA and SpartanXL FPGAs. To use the SelectMAP mode the following signals are used.

D7...D0

The D7 through D0 pins comprise the 8 bit bidirectional data bus.

CCLK

The CCLK is the configuration clock input signal used by the internal configuration logic.

PROG

The PROG input signal resets the internal configuration logic and re-initializes the internal configuration memory.

DONE

The DONE output indicates the completion of configuration and the beginning of the Startup sequence.

INIT

The bidirectional open-drain INIT pin is used to hold off configuration initialization, and it indicates when CRC errors occur in the configuration data.

WRITE

The WRITE input is a write strobe that must be asserted and held throughout the loading of data.

BUSY

The BUSY open-drain output indicates whether the current byte is being loaded or ignored.

CS

The CS input is the Chip Select signal used to enable the FPGA to be sensible of the SelectMAP interface.

M2 M1 M0

The MODE pins M<2:0> must be set to indicate which configuration mode will be used. For SelectMAP configuration these must be <110>.

Configuration Process Steps

The following are the steps for performing a Virtex SelectMAP configuration.

Reset the Configuration Logic

If configuration is to follow power-up, or a recycling of the power source, then the configuration logic will become active already initialized. However, if the FPGA is to be reconfigured without the recycling of power, then the PROG input must be asserted Low for at least 300ns to reset the configuration logic.

Time-out Start of Configuration

After the release of the PROG input, or the powering up of the FPGA, the INIT output stays Low while the internal configuration memory is automatically cleared.

Unlike previous generations of FPGAs, Virtex does not require an additional time-out period following initialization. Configuration may begin as soon as INIT has gone High. The FPGA may ignore as much as the first three clock cycles of data; however, a Virtex Bit-Stream is padded with 8 dummy bytes at the beginning of the data stream to account for this.

Loading Configuration Data

To begin configuration, the WRITE and CS inputs must be asserted Low and held. Each byte is loaded on the rising edge of CCLK. If BUSY was Low during the CCLK transition then the byte was accepted. If it was High then the byte was ignored and must be reloaded on the next clock cycle. BUSY is a synchronous signal; Therefore, CCLK can not be suspended until the BUSY is de-asserted. For configuration clocks below 50MHz the BUSY can be ignored entirely, and removed from the interface design.*

End of Configuration

The DONE output transitions High to indicate the end of configuration and the beginning of the startup sequence. During the startup sequence the D7...D0, WRITE, BUSY, INIT, and CS pins all become user IO. If any of these pins are used by the configured design, then any driving source used only for configuration purposes must be disabled so as not to contend.

Depending on the startup options selected in BitGen, completing the startup phase will require as much as 8 CCLK cycles after DONE has transitioned High. However, since the example shown below uses a free running oscillator, this is not a concern. The Virtex FPGA will complete the startup phase even if the CS signal is de-asserted.

Interface Design

To configure the Virtex FPGA from a parallel PROM, a small interface is needed to generate the PROM addresses and load the data into the FPGA. Although any one of several technologies might be used, a Xilinx CPLD is a simple and cost effective implementation, as shown in Figure 1.

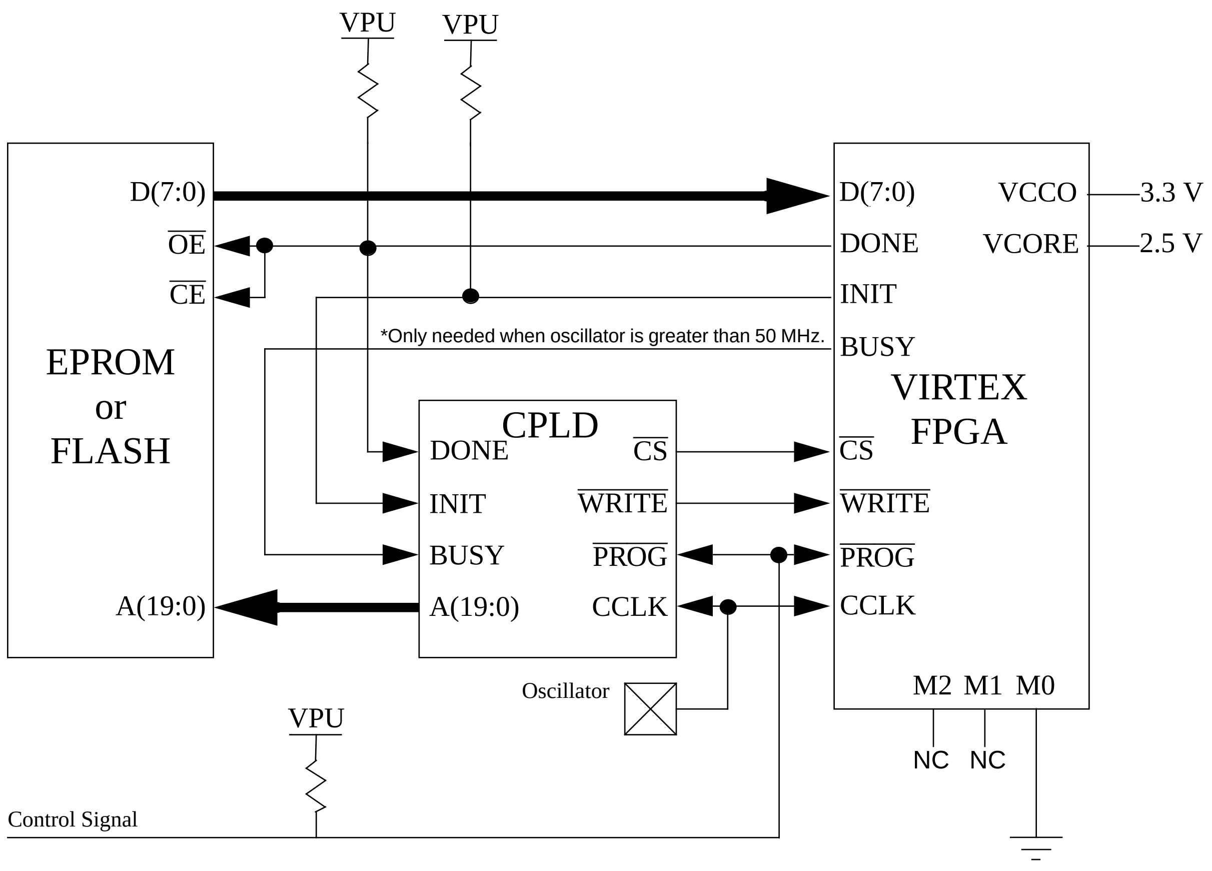

Figure 1: SelectMAP and EPROM Interface Circuit Diagram

Voltage Supplies and Pullups

The Virtex FPGA was designed for mixed voltage environments. The internal voltage (VCORE) requires a 2.5 V supply. During configuration, the IO are automatically set to the LVTTL standard which requires a VCCO of 3.3 V. If this is incompatible with the IO standard to be used after configuration, VCCO may be left disabled during configuration and all the configuration pins (VCCO_2, DONE, INIT, and BUSY) can be pulled up to VPU. Even if the VCCO is driven, the Pullups should remain on DONE and INIT. The VPU refers to the needed IO voltage for the CPLD and PROM. Most likely this will either be 3.3 or 5.0 V. Either is acceptable to the Virtex FPGA. No translation is needed. The recommended pullup value is 4.7K ohm. This value is not critical. Smaller pullup resistors will of course increase current draw when that node is driven to a logic Low.

The PromMAP Design

The CPLD design promMAP, shown in Figure 2, consists of an address counter, a write control register, some tristate buffers and random logic. The PROG input acts as a global reset. The DONE acts as a global tristate. The INIT and BUSY hold off the address counter from incrementing while the FPGA initializes or processes data.

Address Counter

The address counter needs to be large enough to accommodate the address bus width of the PROM. In Figure 1 a 1 Mbyte (8 Mbit) PROM is shown which requires a 20 bit address bus. If less than half of the PROM was to be used, or a smaller PROM in its place, then the address counter could be decreased. For connecting multiple PROMs see "Interfacing Multiple PROMs" on page 4.

Write Control Register

The CS and WRITE input signals to the FPGA must be asserted and held Low for configuration. If only one FPGA is being targeted for configuration these two signals may be derived from the same source. For targeting multiple FPGAs see "Daisy Chaining Multiple FPGAs" on page 5.

Configuration must be delayed until the INIT signal has transitioned High indicating that FPGA initialization has completed.

Oscillator

The Virtex configuration logic was designed to work with a free-running oscillator. The maximum input oscillator frequency for a Virtex FPGA is 66 MHz; however, in this case the maximum frequency for the oscillator is determined by the access time of the PROM (t_{ACC}) plus the setup time for the SelectMAP data inputs (t_{SMDCC}).

$$\text{Oscillator Frequency} = \frac{1}{t_{ACC} + t_{SMDCC}}$$

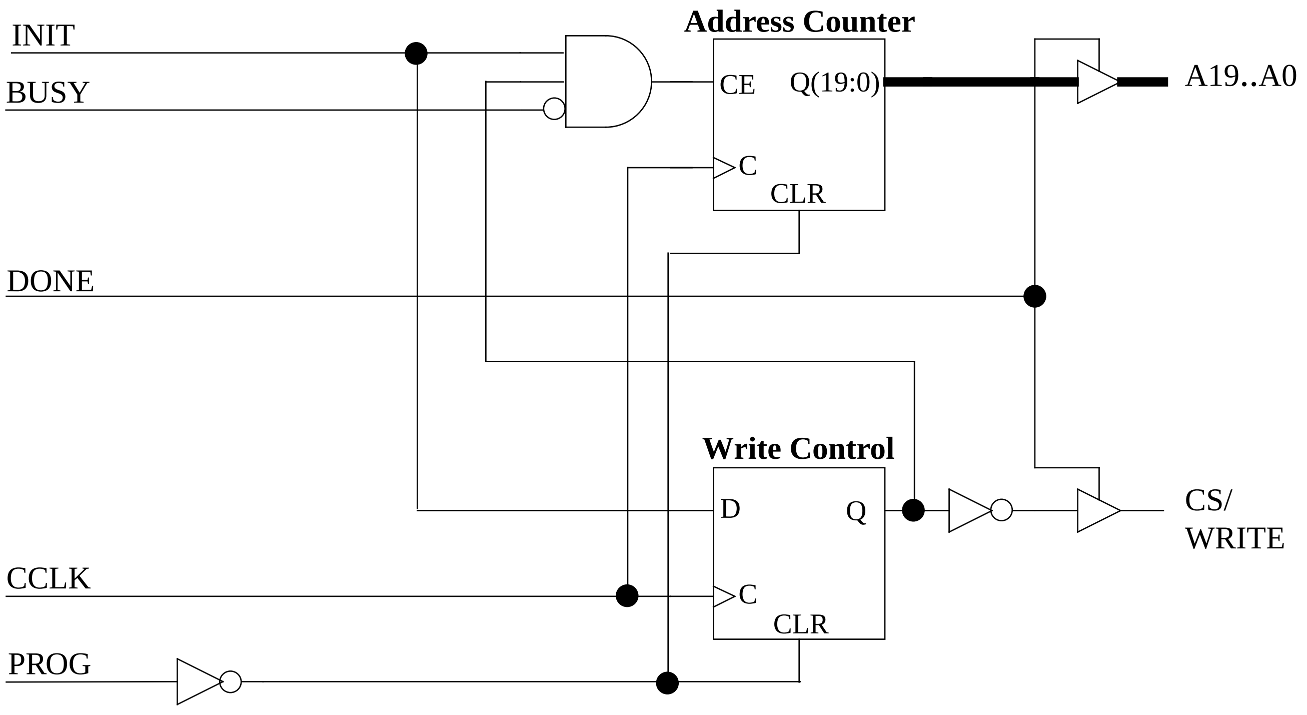

Figure 2: PromMAP Interface Design

The t_{SMDCC} for a Virtex FPGA is 2.0ns. A typical t_{ACC} for an EPROM (e.g. AT27c080) is 100ns. Using these values the maximum frequency of the oscillator is 9.6MHz (76.8 MBit/s). This is far below the CCLK frequency capabilities of the Virtex (66 MHz) and CPLD (66.6 MHz in a XC9536-10). Therefore, a faster PROM would allow for faster configuration speeds up to 66 MHz (528 MBit/s).

PromMAP Design Files

The promMAP design was built in an XC9536VQ44-10 and tested with an XCV300BG432 and an AT27c080. The VHDL, Verilog, source code and Foundation project for the design can be downloaded from the XIlinx website at ftp://ftp.xilinx.com/pub/applications/xapp/xapp137.zip

Interfacing Multiple PROMs

If the needed memory storage capacity requires the use of multiple PROMs then the interface design can be altered to accommodate PROM daisy-chaining as shown in Figure 3.

The output enable (OE) input pins of the PROMs can be driven by an address decode of the most significant bits of the address counter in the promMAP design as shown in Figure 4.

For the example shown in Figure 3, the 8 MBit PROM has been replaced with two 4 MBit PROMs. The common address bus is now only 19 bits wide and the MSB A(19) is used to select between the PROMs, see Table 1.

Table 1: Address decode for two PROM selection

A(19)	OE0	OE1
0	0	1
1	1	0

The CE outputs are just a binary decode of an additional MSB of the address counter. To add more PROMs simply increase the size of the counter and binary decoder.

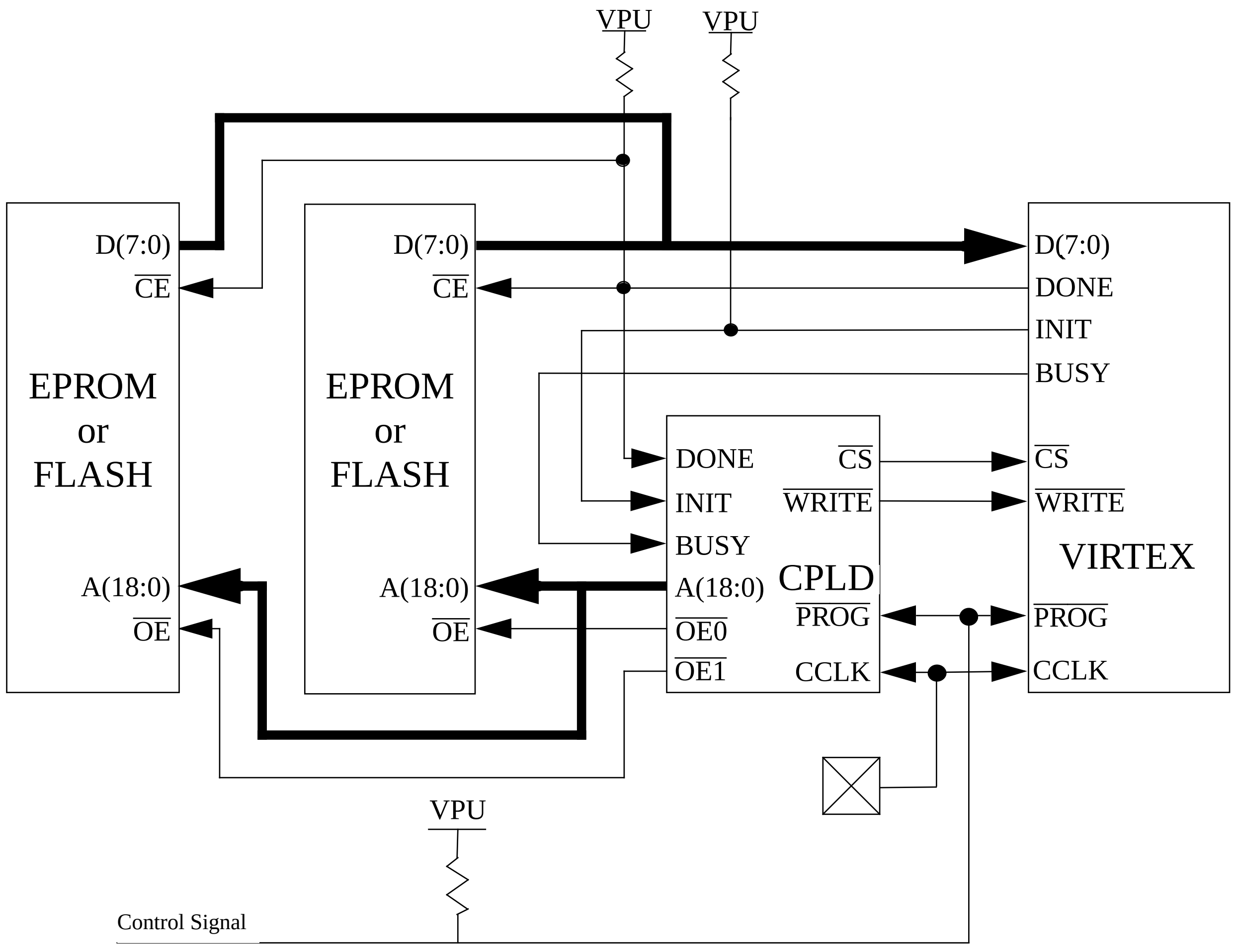

Figure 3: Connecting Multiple PROMs in daisy-chain

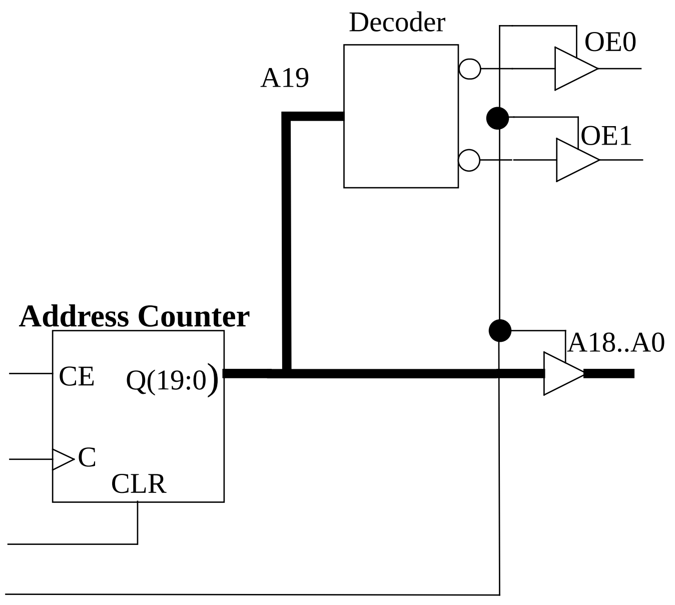

Figure 4: Selecting multiple PROMs

Daisy Chaining Multiple FPGAs

SelectMAP configuration does not support a standard daisy-chain configuration like the bit-serial modes do. However, in an application such as this, where there are multiple FPGAs to be configured, the PromMAP design may be altered to accommodate a parallel daisy-chain so that multiple interface chips are not necessary. As shown in Figure 5, the daisy-chaining of multiple FPGAs in parallel is similar to daisy-chaining multiple PROMs.

For each FPGA the DONE and CS signals must be connected to the interface design separately. The BUSY and WRITE signals can be a common line between the FPGAs. When the first FPGA raises it's DONE signal the interface should disable that FPGA's CS input and enable the next one in the chain, and so on until all DONEs are high. An example of this is shown in Figure 6.

Using this method will cause each FPGA to become active after its DONE transitions High. Therefore, it is important that none of the configuration pins used for SelectMAP configuration be used as IOs in the FPGA designs. Otherwise contention could result when one or more FPGAs are configured and active, but others are not yet configured.

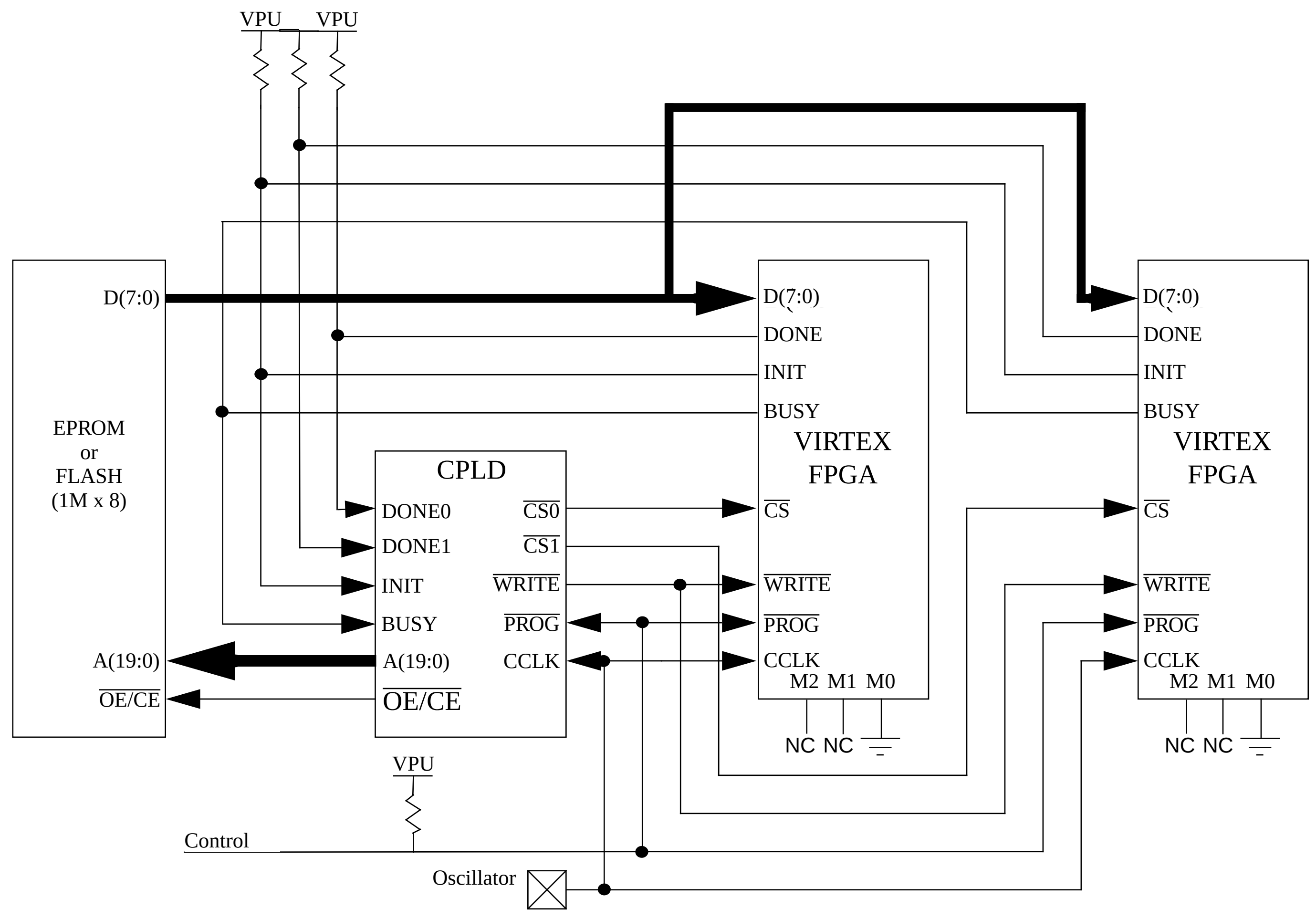

Figure 5: Connecting multiple FPGAs for parallel daisy-chaining

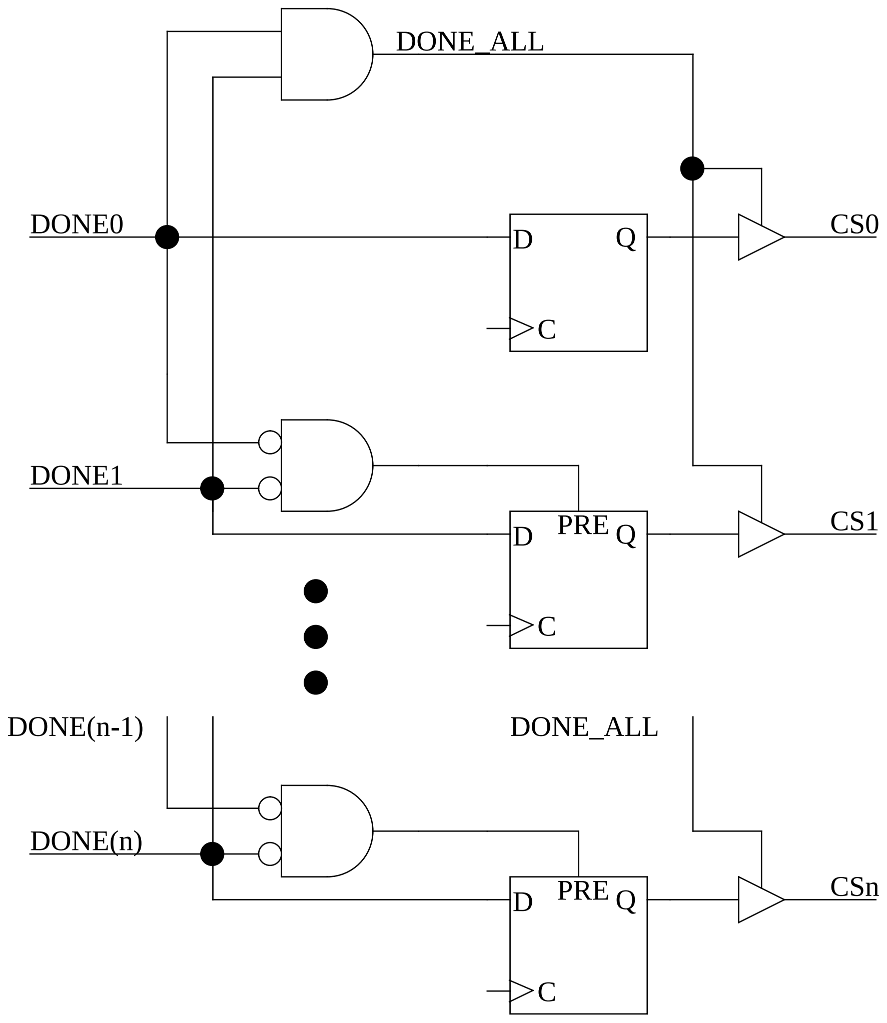

Figure 6: Daisy-chain of FPGA Chip Selects

Alternatively, if it is desired to use these pins after configuration, then the common WRITE signal input for each FPGA could be connected to the GTS and GSR (or other reset signal) in the FPGA design so that these pins will not be driven by the active FPGAs while others are still configuring.

Figure 6 shows how the CS outputs are cascaded accordingly with the DONE inputs. In the single FPGA example the DONE input is the global tristate control for all outputs. For multiple FPGAs the global tristate signal (DONE_ALL) is generated by ANDing together all the DONE lines from the separate FPGAs.

The dotted extension in Figure 6 demonstrates that more FPGAs may be added to the chain by adding to the chain of chip select registers. For each segment, the register is held asynchronously PREset when the previous ordered DONE (n-1) and the associated DONE (n) are both Low. When the previous ordered DONE signal transitions High, the PREset is released and the register loads the value of the associated DONE signal on each clock cycle. When the associated DONE transitions High, the corresponding CS output will transition High disabling the configuration for that FPGA in the chain. This removes the PREset of the next segment and allows configuration of the next FPGA in the chain. The first segment does not make use of a PREset. When all DONE signals are High all outputs will disable.

The WRITE signal need only be held Low throughout the entire configuration process and disabled when DONE goes High. Since the CS selects will not assert and the address counter will not increment until INIT is High, the WRITE signal input to the output tristate buffer may simply be a GND connection (a true Open-Drain style output).

Synchronizing Startup

An alternative method to activating the FPGAs sequentially would be to synchronize the startup of all the FPGAs by combining all the DONE lines into one signal. Virtex automatically synchronizes the startup sequence to wait for DONE to be externally released. It is, however, important that the BitGen option to "DriveDone" is not selected. By default this is not the case. Unless otherwise specified, the DONE pin will be an open-drain output.

Combining the DONE signals means they cannot be used to indicate when configuration must transition to the next FPGA in the chain. Instead address decoding would be used to disable and enable the individual CS outputs.

In Figure 7, decoding logic PREsets the CS control register at Address 0 (set all CS registers High in the beginning) and Toggles the register at the starting address and ending address for the desired configuration data. Set the starting address of the first register to 1h. Otherwise, the register will not toggle. The decode of the starting address for one stage can also serve as the ending address decode of the previous stage.

Synchronizing Startup in this manner simplifies the board connections and worries of contention during configuration. However, this also increases the size of the PromMAP design. The starting and ending address values for each stage is easily determined after generating the final PROM file.

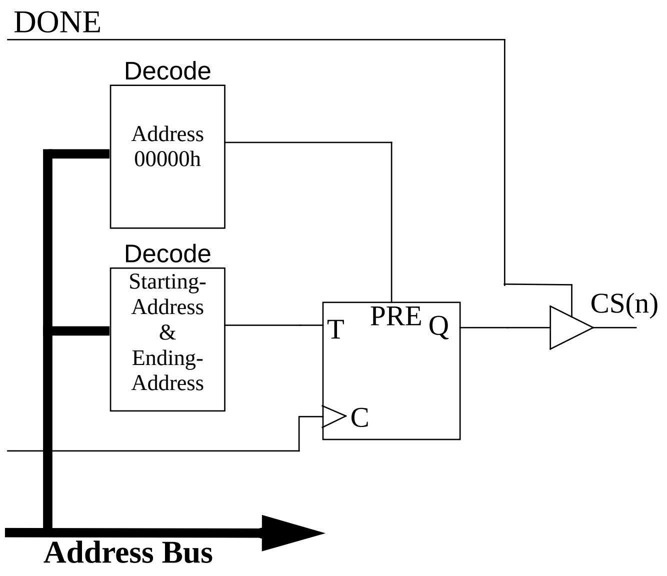

Figure 7: CS Address Decoding

Cascading PROM Files

PromGen must be run on each FPGA design separately and the resulting prom files concatenated together. Care must be taken to assure that the prom files are in the desired order. Do not use PromGen to create a single prom file as would be done for bit-serial daisy-chaining. This would result in an incorrect bit stream for this configuration scheme.

The PROM files for each of the FPGAs in the daisy-chain must be simply concatenated together. This may be accomplished in a variety of ways depending on the programmer used.

If your programmer can use HEX files then a simple concatenation of hex files is all that's needed. The PromGen utility has a switch option to create HEX files.

If your PROM Programmer's software has edit ability, then it may allow you to read in each of the PROM files individually and specify which address to load up from.

If the programmer requires a single PROM file in addressed format (mcs, tek, exo), then some manual editing may be required. It should be noted that a daisy chain of Virtex PROM files will typically be too big for the MCS and TEK formats. EXO is usually the better choice and easier to work with.

To concatenate EXO files together, first generate the EXO for the first bit file in the chain with PromGen loading up from address 0h. View the EXO file. The very last line should be deleted. The second to last line should also be deleted, but first note the starting address for that line. For example if the last 2 lines read:

S20803576000000003D

S804000000FB

What you see here is that the data 00000000 is loaded in starting at address 35760h. This line is only 4 bytes, a partial line (a full line is 16 bytes), this line should be deleted and the next prom file generated should start at address 35760h. Then the two files may be concatenated together. For more stages to the daisy chain, simply continue with this same process.

The 4 bytes of data that is being removed from the PROM file are at the end of a 16 byte section of dummy data that merely represents needed clock cycles to complete the startup sequence (actually, only 8 clock cycles are needed). Since a free running oscillator is being used in this case, the unwanted dummy bits may be discarded.