XAPP354 (v1.1) September 30, 2002

Using Xilinx CPLDs to Interface to a NAND Flash Memory Device

Summary

This application note describes the use of a Xilinx CoolRunner™ CPLD to implement a NAND Flash memory interface. CoolRunner CPLDs are the lowest power CPLD available and the ideal target device for memory interface applications. The code for this design may be downloaded from the Xilinx web site, refer to section **HDL Code**, page 14. This design fits XCR3032XL CooRunner or XC2C32 CoolRunner-II CPLDs.

Introduction

The flash memory market has been rapidly growing over the past several years due in part to increasing demands of portable and embedded devices. This market is driven by embedded code storage and bulk data storage applications. Flash technology has been optimized to meet the needs of these two target applications.

NAND Flash is a sequential access device appropriate for mass storage applications, while NOR Flash is a random access device appropriate for code storage applications. NAND technology organizes cells serially to achieve higher densities. This reduces the number of contacts needed in the memory array. The trade-off between the two technologies is NAND Flash data must be accessed sequentially compared with NOR Flash which offers fast random access.

NAND Flash memory offers low cost per bit, high performance, and the highest density non-volatile memory available. NAND Flash is ideal for applications ranging from MP3 players and digital cameras to applications requiring mass storage of data, especially when the data is packetized, or sequentially arranged.

This application note describes the design of a basic NAND Flash interface. The NAND devices used for testing this interface include the AMD UltraNAND™ Flash and the Samsung NAND Flash memory.

The AMD UltraNAND flash memory (AM30LV0064D) is a 64 Mbit storage device. This memory device utilizes a multiplexed command/data/address bus as well as other control signals for read, erase and program commands. This memory device has an initial page read access time of 7 µs, with subsequent byte access times of less than 50 ns per byte.

The Samsung K9F4008W0A is a 512K x 8-bit NAND Flash memory device. The command, data, and address are multiplexed through an 8-bit I/O port. This memory device supports a 32-byte frame read, with random access times of 15 µs and sequential access times of 120 ns.

NAND Interface

The NAND interface described here is implemented in a CoolRunner CPLD. The NAND interface design can interact with both AMD and Samsung NAND memory devices.

Figure 1 shows the overall system diagram for a single AMD UltraNAND Flash or Samsung memory device. The CoolRunner CPLD reads the 4 least significant bits in the system address

in order to decode the flash interface commands. The interface signals to the Flash device are asserted by writing to a specific port of the CPLD.

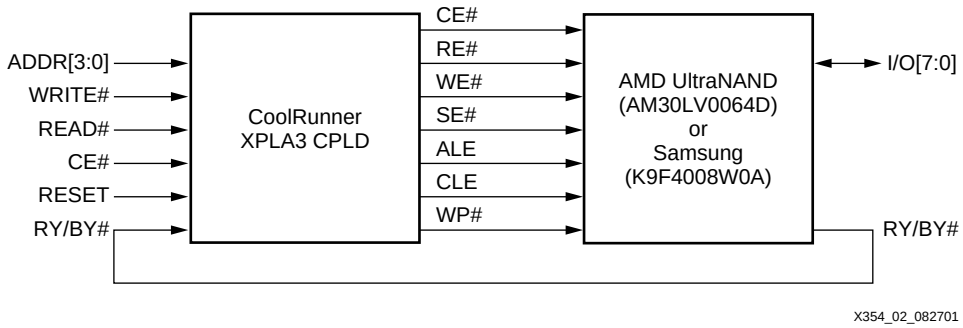

Figure 1: **System Block Diagram**

The NAND Flash interface signals and functionality is shown in Table 1.

Table 1: **UltraNAND Pin Descriptions**

Pin Name	Function
I/O[7:0]	I/O pins used to send commands, address, and data to the device, and receive data during read operations.
CLE	Command Latch Enable. The CLE input controls writing to the command register. When CLE is high, the command is loaded on the rising edge of WE#.
ALE	Address Latch Enable. The ALE input controls writing to the address register. When ALE is high, the address is loaded on the rising edge of WE#. ALE must remain high during the entire address sequence.
CE#	Chip Enable. The CE# input controls the active vs. standby mode of the device. During a command or address load sequence, CE# must be low prior to the falling edge of WE#.
RE#	Read Enable. The RE# input controls the data and status output on the I/O lines. The data output is triggered on the falling edge of RE#.
WE#	Write Enable. The WE# input controls the data and command on the I/O lines during a write sequence. The I/O lines are latched on the rising edge of the WE# signal.
WP#	Write Protect. The WP# input provides protection when programming or erasing the device. The internal voltage regulator is reset when WP# is low, preventing any program or erase operations.
SE#	Spare Area Enable. The SE# input controls access to the 16 bytes of spare area on each page. When SE# is not asserted (high), the spare area for the selected page is not enabled. When SE# is asserted (low), access to the spare area is enabled.
RY/BY#	Ready/Busy Output. The RY/BY# output indicates the operation status of the device. When RY/BY# is high, the device is ready for the next operation. When RY/BY# is low, an internal program, erase, or random read operation is in progress.

AMD UltraNAND Memory Device

The AM30LV0064D is organized as 8 kB (+ 256 byte spare area) blocks (1,024 blocks total). Each block has 16 pages of 512 bytes (+ 16 bytes spare area) or 16,384 pages total. Figure 2 is a block diagram of the AMD UltraNAND device.

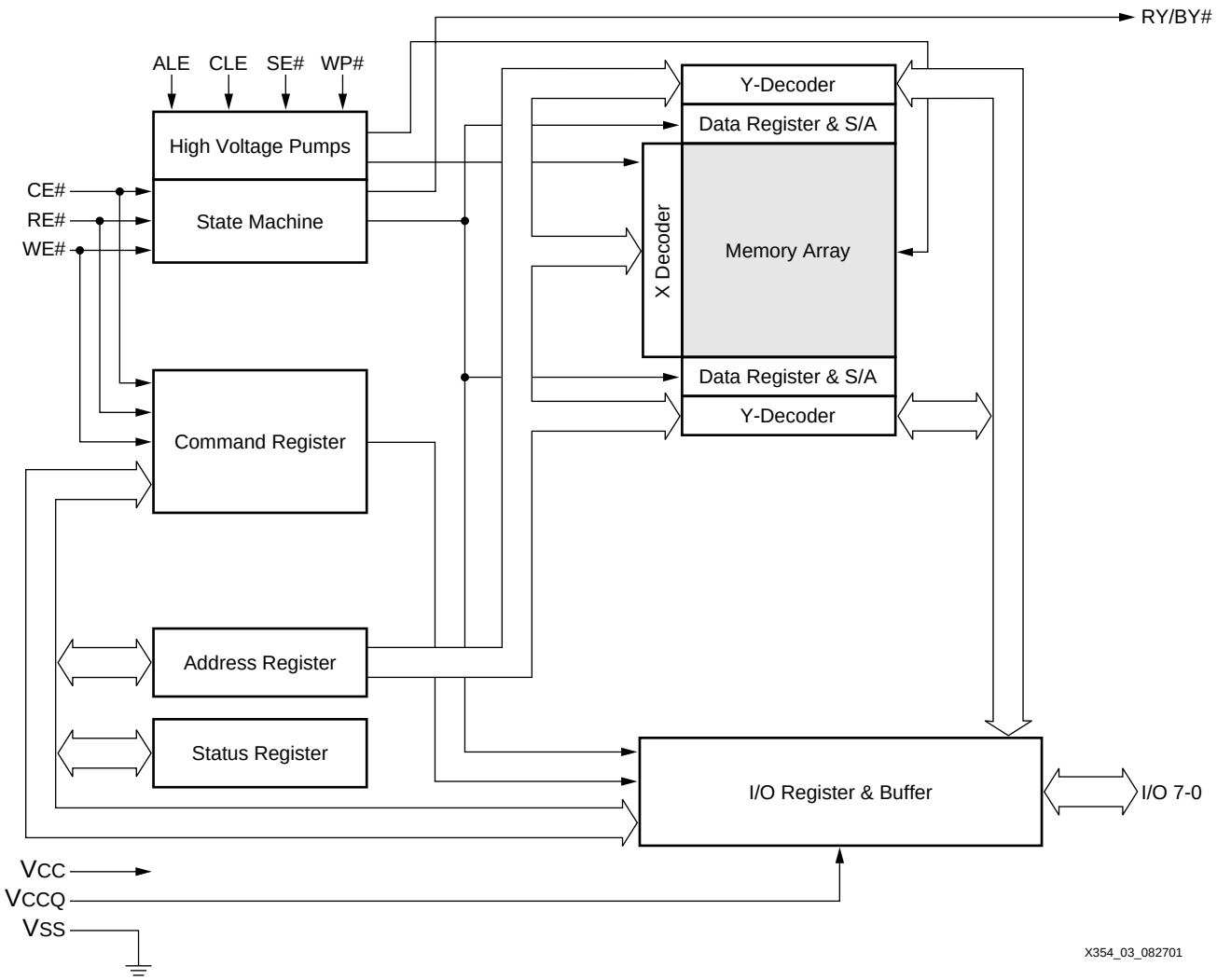

Figure 2: **AMD UltraNAND Block Diagram**

Table 2 describes the AMD UltraNAND command set and functionality. The command register does not occupy any addressable memory location. This register holds the command, along with any address and data information needed to execute the command. Programming data into the Flash array is a two step process. The data to be programmed is loaded into the data

registers using the "Input Data" command. After the data is loaded, the "Page Program" command is performed to write data from the data registers to the Flash array.

Table 2: **AMD UltraNAND Command Set**

Operation	Cycle (Hex)	Functionality
Read Data	00h or 01h	Reads out flash array starting with First Half Page (00h) or reads out data starting with the Second Half Page (01h).
Gapless Read	02h	Allows reading from multiple pages with only one 7 µs latency occurring on the first page transfer
Read Spare Area	50h	Only reads data from the 16 byte spare area in each page (address locations 512 through 527).
Read ID	90h	Read the manufacturer and device ID.
Read Status	70h	Checks the device status to determine if the device is ready, in the write protect mode, erase suspended, or if the previous program/erase operation completed without error.
Input Data	80h	First command that allows the device to be programmed. This command loads the data registers from the I/O lines to program the device.
Page Program	10h	Issued after the "Input Data" operation has loaded the proper data. The command transfers information from the data registers to the Flash array in 200 µs or less, and the Flash device will appear busy during the data transfer operation.
Block Erase	60h & D0h	In the first command (60h), two address cycles are used to input the address of the block to erase. Once the second command (D0h) is issued, the Flash device will begin the "Block Erase" operation.
Erase Suspend	B0h	Only valid during a "Block Erase" command sequence. On the rising edge of WE#, the erase operation will be suspended.
Erase Resume	D0h	Only valid during an "Erase Suspend" command sequence. On the rising edge of WE#, the Flash device will resume the "Block Erase" operation.
Reset	FFh	Used to initialize the Flash device.

Samsung NAND Memory Device

The K9F4008W0A is a 512K x 8-bit NAND Flash memory. A Program operation programs a 32-byte frame in typically 500 µs and an Erase operation erase a 4 kB block in typically 6 ms. Data in a frame can be read out at a burst cycle rate of 120 ns/byte. The array organization of the Samsung device is such that 32 bytes are equal to one accessible frame. A row consists of four frames (or 128 bytes) and one block consists of 32 rows (or 4 kB). The total size of the device

is 128 blocks (or 4 MB). Data is programmed via the Frame Register which holds 32 bytes of data. Figure 3 is a block diagram of the Samsung K9F4008W0A.

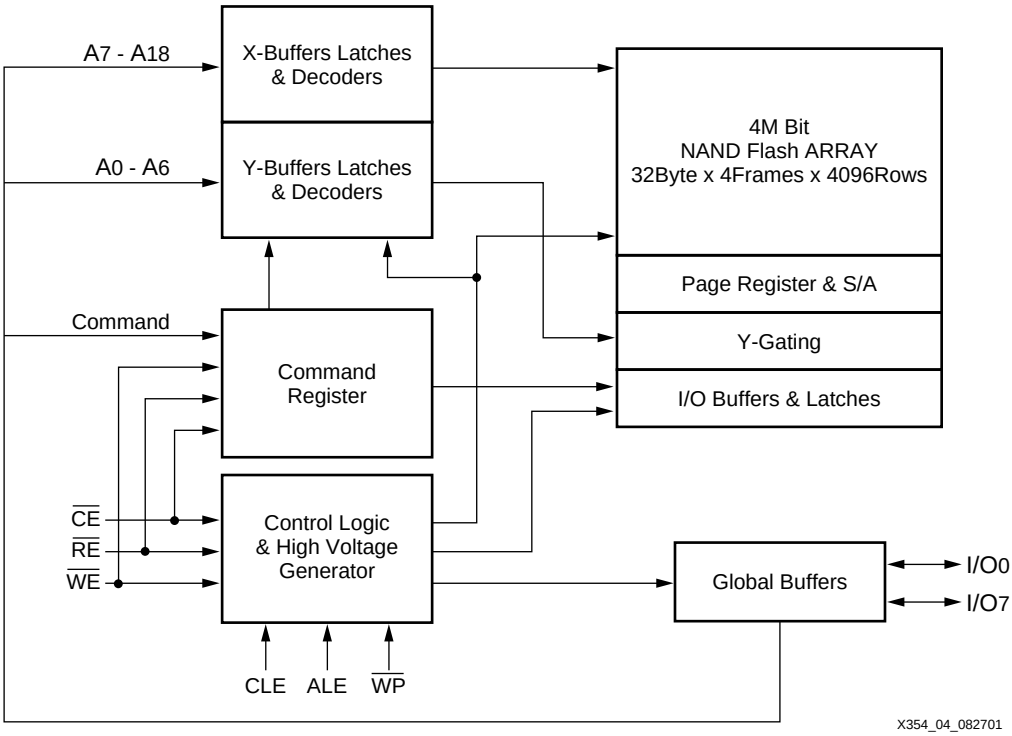

Figure 3: **Samsung Block Diagram**

All address and command instructions are multiplexed through the 8 I/O lines. Similar to the AMD UltraNAND, data is latched on the rising edge of WE# when CE# is low. The address is latched in when ALE = '1' and CLE = '0' and the command is latched when ALE = '0' and CLE = '1'. Table 3 describes the Samsung NAND Flash memory command set.

Table 3: **Samsung Command Set**

Command	Cycle Data	Description
Read	00h	Device default mode. After the frame address is changed, 32 bytes of data are transferred to the data registers in less than 15 µs. Each data byte in the frame can be read on the high to low transition of the RE# signal.
Reset	FFh	Reset operation can abort a read, program, or erase operation. The device enters the Read mode after a Reset command.
Frame Program	80h & 10h	Frame Program consists of loading the 32 byte Frame Register and a nonvolatile programming period when the data is programmed into the appropriate cells.

Table 3: **Samsung Command Set**

Command	Cycle Data	Description
Block Erase	60h & D0h	The Erase operation is done 4K bytes (or 1 block) at a time. Requires a 2 cycle address load to specify block address (A_{18} to A_{12}).
Status Read	70h	The device contains a Status Register which may be read to determine if a program or erase operation is complete and successful. See the Samsung data sheet (refer to **References**, page 15) for a complete definition of each bit in the Status Register.
Read ID	90h	The device contains product identification, which can be read during a Read ID command. Two read cycles are required to read the manufacturer code and device code.

CPLD Design

For more information on the ABEL implementation of the CoolRunner CPLD design, refer to AMD Application Note # 22363 (see **References**, page 15). The design described here and available on the web is implemented in both VHDL and Verilog (see "HDL Code" on page 14 for more information).

The CPLD design decodes system address commands to interface with the AMD UltraNAND and Samsung Flash memory devices. The CPLD is responsible for the following functions:

- Decode read or write from address bus
- Interpret system address bus commands
- Assert interface signals to UltraNAND Flash device
- Monitor RY/BY# output from Flash memory device

The CPLD is configured to decode address 00h through 0Fh from a base address. The CPLD logic outputs are determined by the system writing to each port address. The port addresses for the CPLD are shown in Table 4 with a functional description of each.

Table 4: **CPLD Port Address Definition**

Port	Operation	Function
0	Read Data/ID/Status or Write Address/Data	Read information from previous command loaded. Write address (ALE high) or data (ALE low).
1	Command	All commands are written through this port with ALE low.
2	Set ALE	Set ALE (high) to allow addresses to be written.
3	Clear ALE	Clear ALE (low) to allow commands or data to be written.
4	Set SE#	Set SE# (low) to allow access to the spare area on each page.
5	Clear SE#	Clear SE# (high) to prevent access to the spare area on each page.
6	Set WP#	Set WP# (low) to prevent program/erase cycles.
7	Clear WP#	Clear WP# (high) to allow program/erase cycles.
8	Set CE#	Set CE# (low) to enable the UltraNAND device.
9	Clear CE#	Clear CE# (high) to disable the UltraNAND device.

Table 4: **CPLD Port Address Definition**

Port	Operation	Function
A	N/A	Not used in this design.
B	N/A	Not used in this design.
C	N/A	Not used in this design.
D	N/A	Not used in this design.
E	N/A	Not used in this design.
F	RY/BY# Status	Read the state of all RY/BY# pins through this port.

Table 4 is described in more detail in the AMD Application Note #22363 for the design implementation in ABEL. Refer to **References**, page 15 for more information.

Figure 4 is a block diagram of the VHDL/Verilog implementation of the NAND interface. All port signals (shown in Table 4) are decoded from the system address when CE# is asserted.

The ALE_SIG process asserts the ALE signal on a write to PORT2 and clears ALE on a write to PORT3. The SEN_SIG process asserts the SE# signal upon a write to PORT4 and clears SE# upon a write to PORT5. The OUTCE_SIG process asserts the OUTCE# signal upon a write to PORT8 and clears OUTCE# upon a write to PORT9. The WPN_SIG process asserts the WP# signal upon a write to PORT6 and clears WP# upon a write to PORT7. The READY_SIG process assigns the RY/BY# signal from the Flash device to the ready output signal. Otherwise, the ready signal is 3-stated.

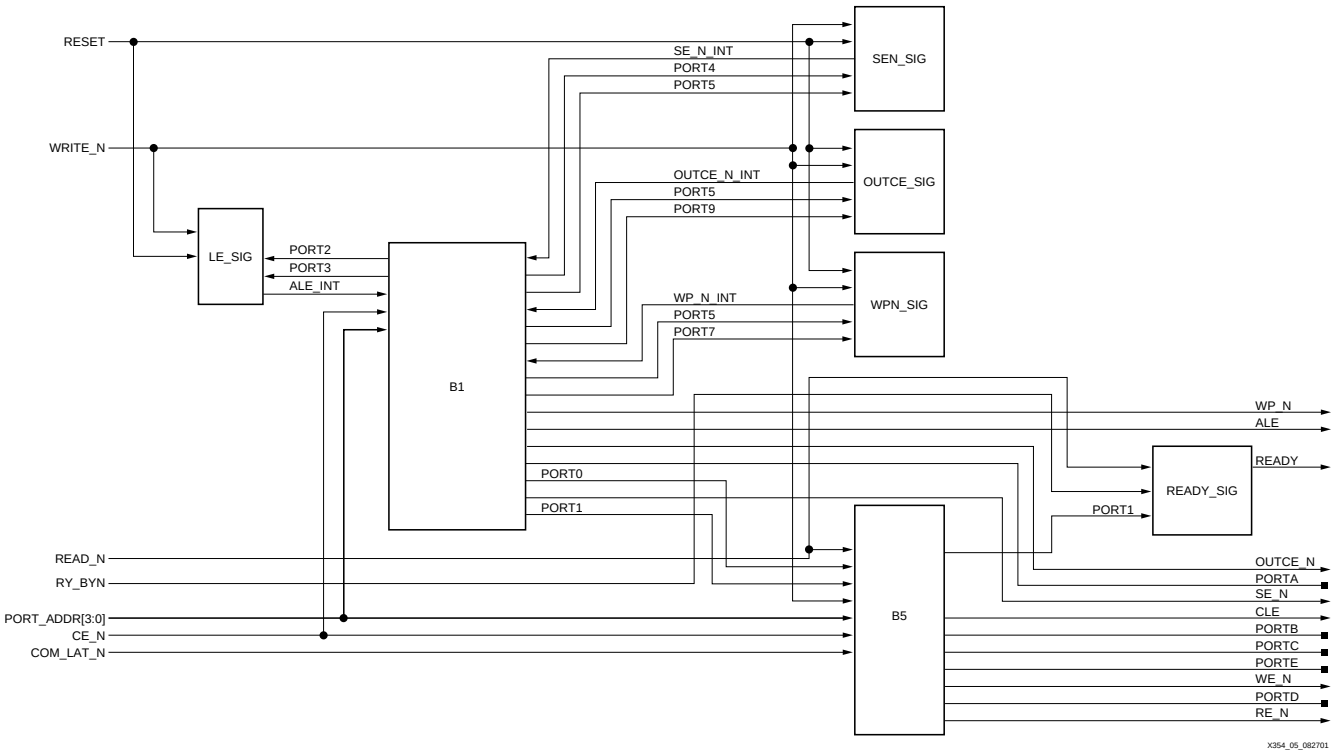

Figure 4: **CPLD Block Diagram**

The CLE signal is asserted on any access to PORT1. The RE# signal is asserted to the Flash device when a read is performed on PORT0. The WE# signal is asserted to the Flash device when a write is performed on PORT0 or PORT1.

CPLD Implementation

The NAND flash interface was implemented in VHDL and Verilog as described above; and in ABEL as described by AMD. Xilinx Project Navigator was used for design compilation, fitting, and simulation in a CoolRunner XPLA3 CPLD. Xilinx Project Navigator, which includes the ModelTech simulation tool, is available free-of-charge from the Xilinx website at www.xilinx.com/products/software/webpowered.htm. The design was targeted for a 3.3V, 32 macrocell CoolRunner XPLA3 CPLD (XCR3032XL-VQ44). The design utilization is shown in Table 5. This utilization was achieved using certain fitting parameters, actual results may vary.

Table 5: **NAND Interface XPLA3 32-Macrocell Utilization**

Resource	Available	Used	Utilization
Function Blocks	2	2	100%
Macrocells	32	10	31.25%
Product Terms	96	47	48.96%
I/O Pins	32	21	65.63%

Design Verification

Verification of the NAND interface was performed using the Xilinx WebPACK Project Navigator VHDL output timing model. The timing model was imported and compiled by Model Technology ModelSim. A test bench was utilized to instantiate the memory device (AMD UltraNAND or Samsung Flash) and the CPLD interface design as shown in Figure 5. The AMD UltraNAND and Samsung devices were instantiated using Denali Memory Maker.

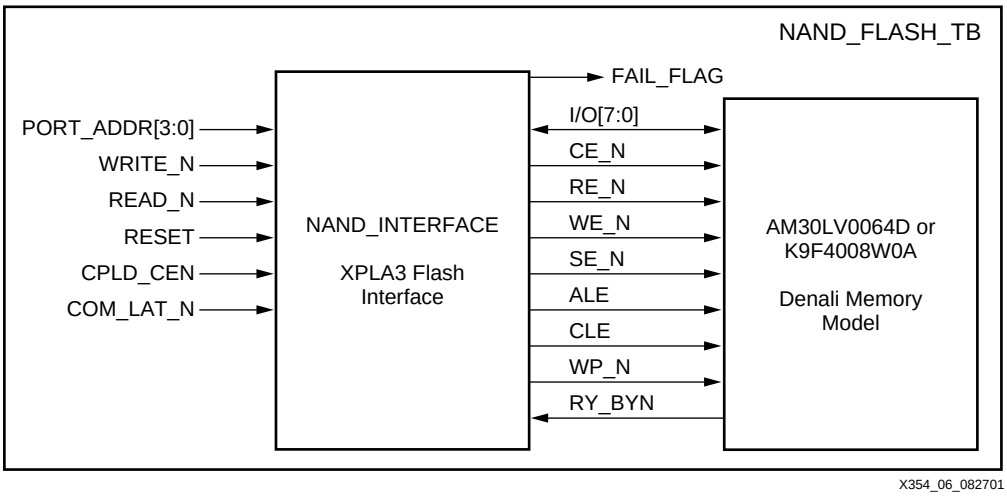

Figure 5: **Test Bench Diagram**

Figure 5 illustrates the operational flow used to test the Denali memory model. Figure 6 describes the test flow for the AMD UltraNAND device. The test flow for the Samsung Flash memory model was modified slightly to match Samsung command cycles.

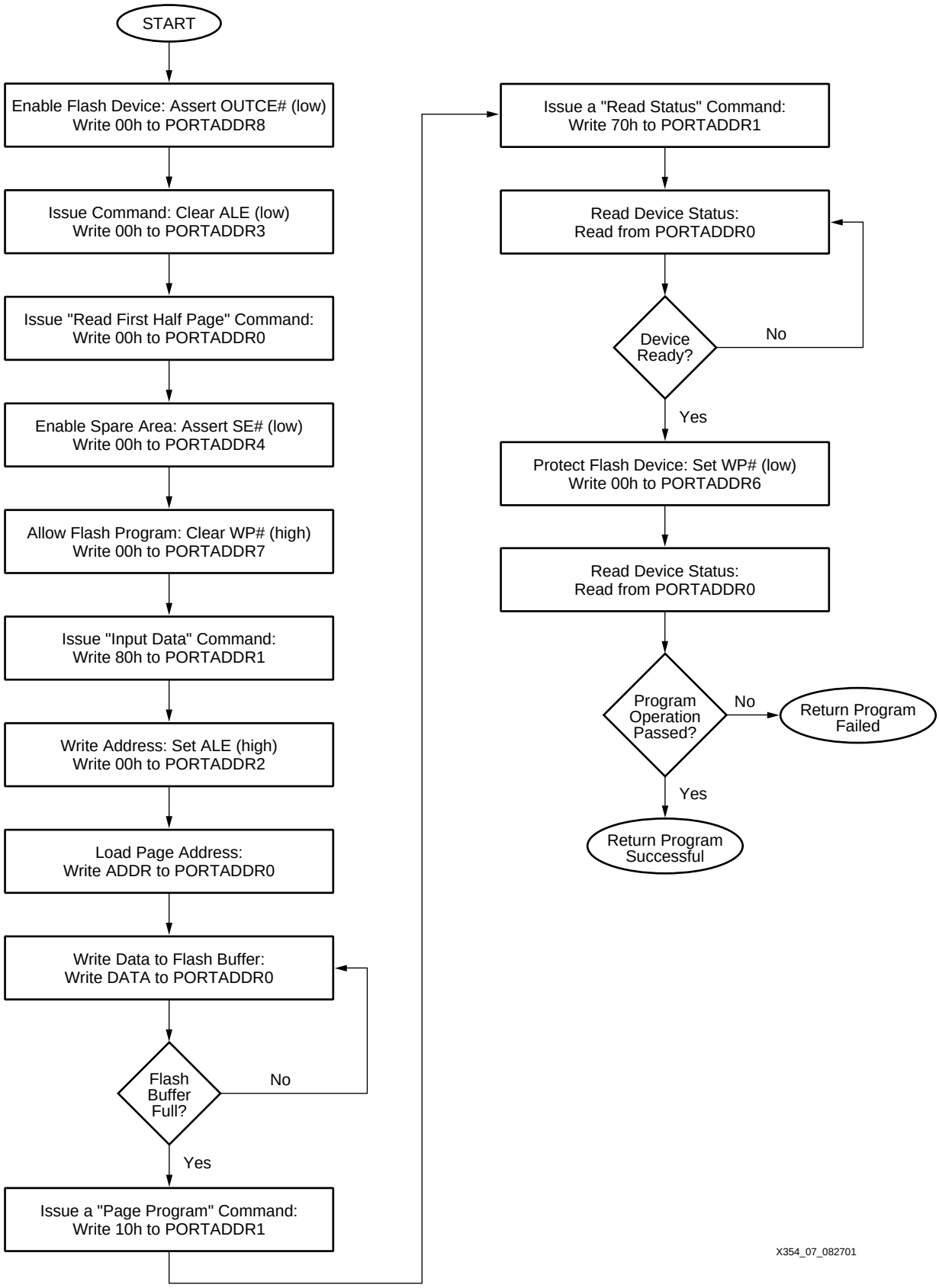

Figure 6: **Memory Operational Test Flow**

Denali Memory Maker

The Denali Memory Maker tool is used to simulate the AMD UltraNAND and Samsung devices. The Denali tool allows a VHDL or Verilog model to be instantiated in a test bench environment. In this simulation, Model Technology ModelSim is the target simulator. For a given memory device, a SOMA file (Specification Of Memory Architecture) represents unique timing, features, and functionality. The SOMA file is then imported to the Denali MemMaker tool. The SOMA file can be edited to meet the design signal names and timing requirements. Figure 7 illustrates how to bring a SOMA file into the MemMaker tool.

Figure 7: **Opening a SOMA File in MemMaker**

Figure 8 illustrates how to change any signal name requirements in the MemMaker tool.

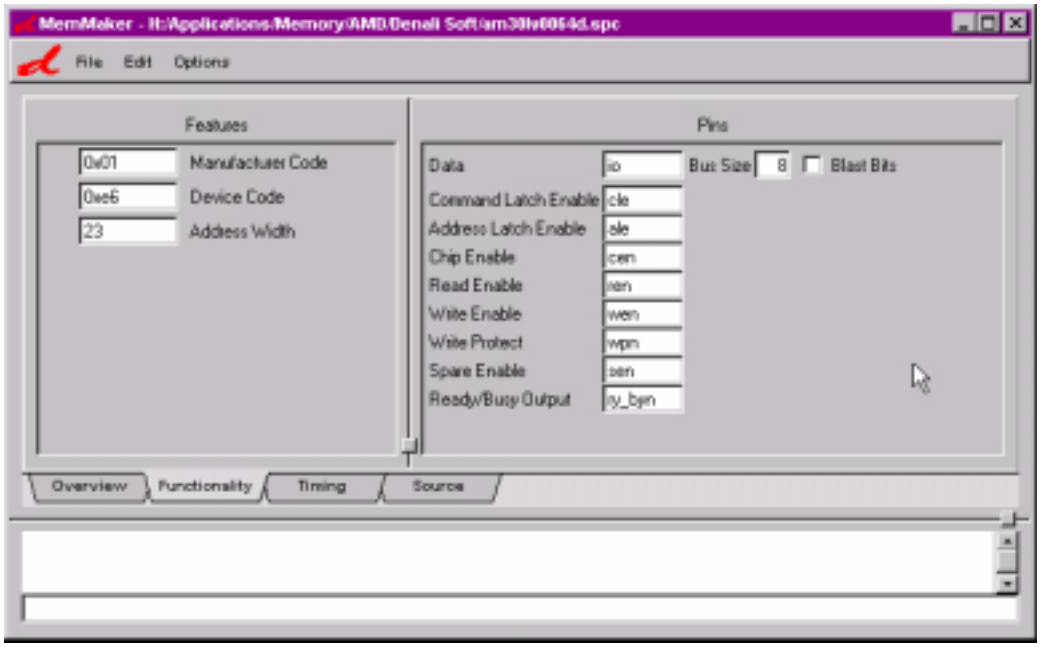

Figure 8: **MemMaker Functional View**

Once all signal name and timing requirements have been specified, the VHDL or Verilog source code can be generated. To generate VHDL source code, select Options | Simulation

Environment | VHDL | Model Technology ModelSim (Windows). The VHDL code can then be saved for use in a test environment as shown in Figure 9.

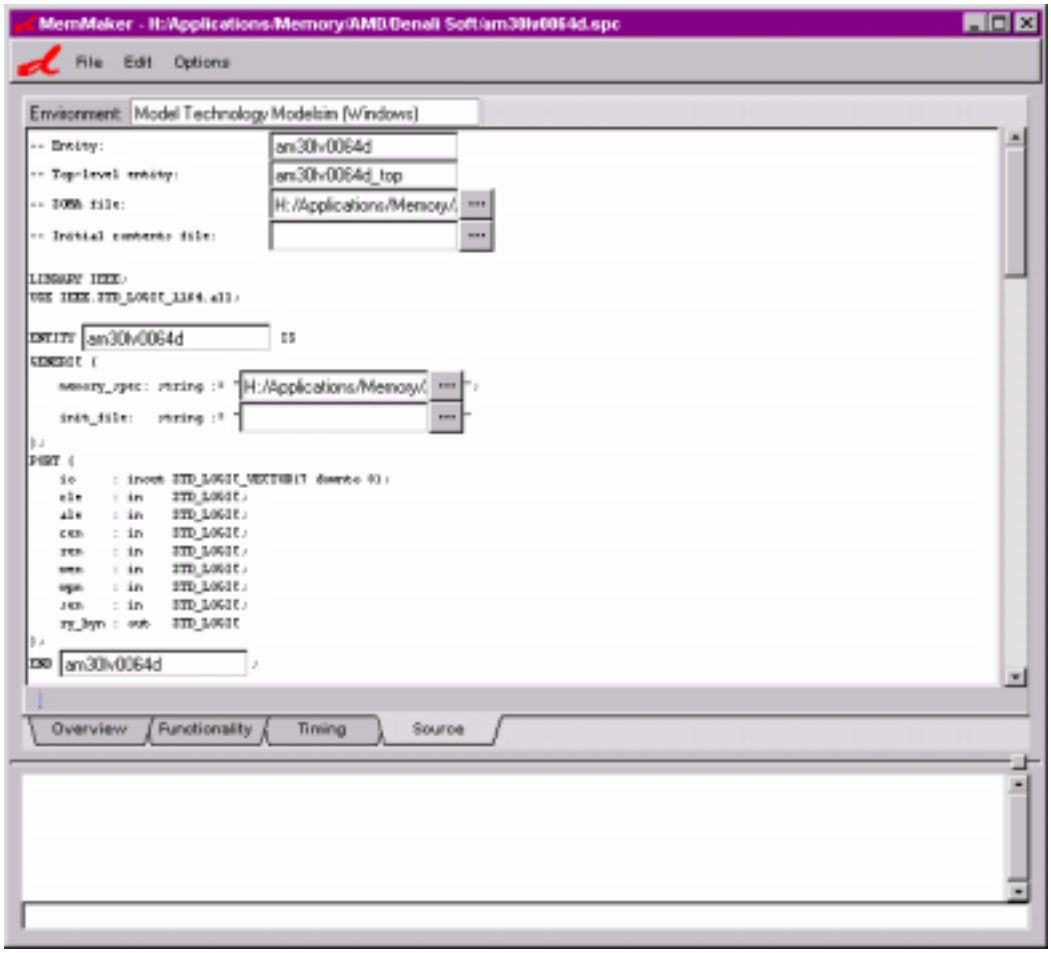

Figure 9: **MemMaker Source Code View**

ModelSim Implementation

Note:

Please refer to XAPP338: Using Xilinx WebPack and ModelTech ModelSim Xilinx Edition as a guide to using ModelSim with Project Navigator. The ModelSim Quick Start demo provides a good first step for getting familiar with ModelSim.

Model Technology ModelSim was the target simulator in this design. The test bench created for the CPLD design generates the necessary system address cycles. A separate test bench environment was used to test the AMD UltraNAND device and the Samsung Flash memory. This is due to the data buffer size during a program cycle and differences in command codes. The Denali MemMaker model is loaded in ModelSim as illustrated by the ModelSim script.

```
vcom -reportprogress 300 -work work{../amd_flash_tb.vhd}
# Model Technology ModelSim XE vcom 5.3d Compiler 2000.03 Mar 30 2000
# -- Loading package standard
# -- Loading package std_logic_1164
# -- Loading package numeric_std
# -- Loading package pkg_convert
# -- Compiling entity amd_flash_tb
# -- Compiling architecture behavior of amd_flash_tb
# -- Loading entity am30lv0064d
# -- Loading package pxa_pkg
# -- Loading entity nand_int
vsim work.amd_flash_tb
```

```
# vsim work.amd_flash_tb
# Loading C:/Modeltech_xe/win32xoem/../std.standard
# Loading C:/Modeltech_xe/win32xoem/../ieee.std_logic_1164(body)
# Loading C:/Modeltech_xe/win32xoem/../ieee.numeric_std(body)
# Loading work.pkg_convert(body)
# Loading work.pxa_pkg
# Loading work.amd_flash_tb(behavior)
# Loading work.am30lv0064d(behavior)
# Loading C:\Denali\denali.dll
# *Denali* History enabled for Denali Memory Modeler.
# *Denali*     Debug information will also be saved.
# *Denali* Trace function enabled for Denali Memory Modeler.
# *Denali* Denali Memory Model Version 2.900-0005
# *Denali* Copyright (c) Denali Software, Inc., 1996-2001, All Rights
Reserved.
# *Denali* Class: flash_nand  Instance: "/den_model" Size: 8192Kx8
# *Denali* Class: internal  Instance: "/den_model(spare)" Size: 256Kx8
# Loading work.nand_int(structure)
# Loading work.pxa_bufif2(behavioral)
```

AMD UltraNAND Flash Memory

The test bench for the AMD UltraNAND Flash memory executes the following commands: "Page Program", "Read Status", "Read First Half Page", and "Input Data" as described in Figure 6. In executing these commands, the test bench fills a 528 byte buffer and programs the target page in the UltraNAND device. The status of the operations are checked through the "Read Status" operation.

Figure 10 illustrates loading the page address to the UltraNAND. Notice the ALE signal is high and WE# is asserted for each portion of the address write.

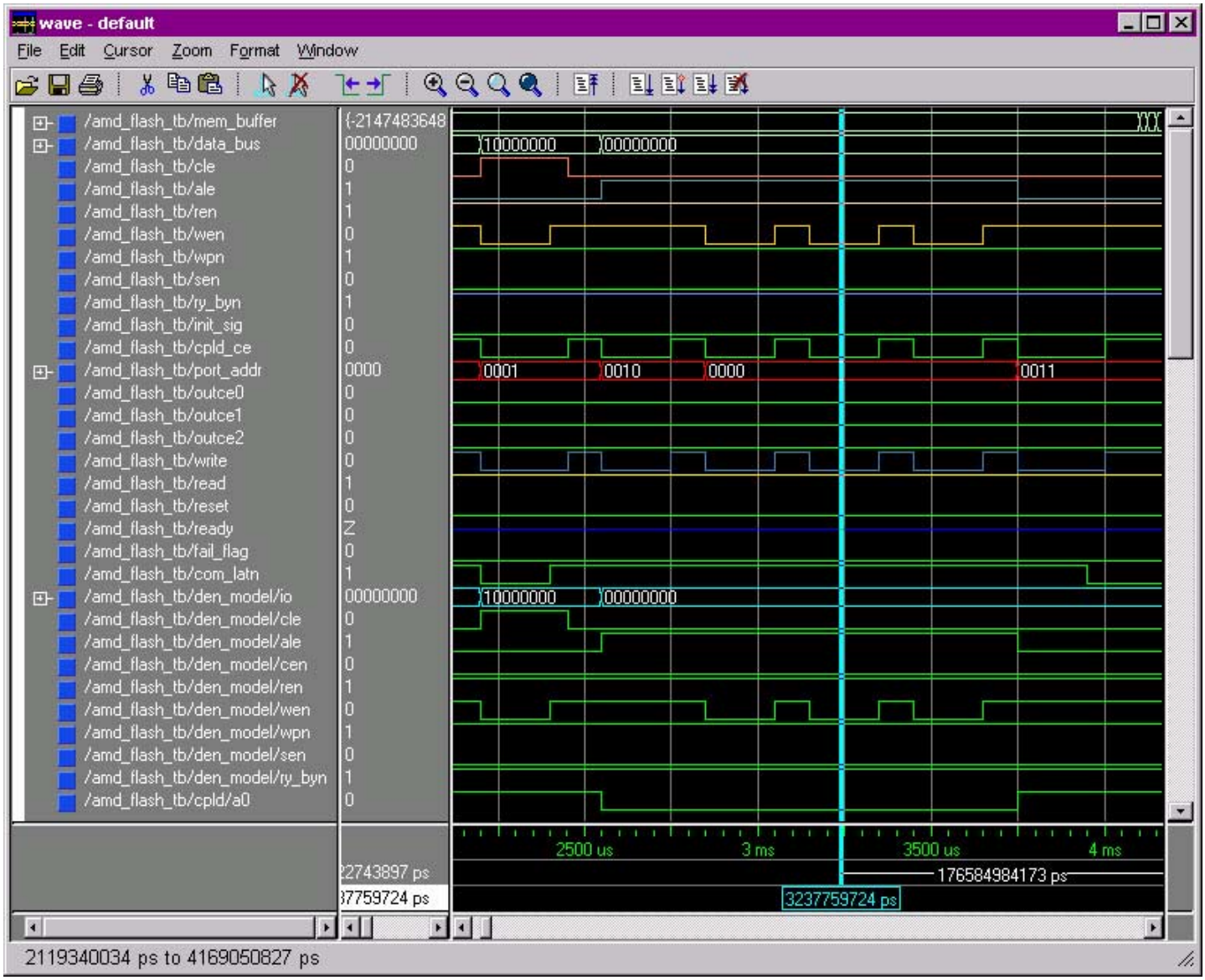

Figure 10: **Memory Address Load**

Figure 11 illustrates the completion of the "Page Program" command. Notice the RY/BY# signal is asserted, representing that the flash device is busy with an operation. After the "Page Program" command is sent to the flash, a "Read Status" command is sent. The "Read Status" command is used to read the device status. When I/O6 is equal to '1', the device is ready for the next command. To check if an operation is successful, reading I/O0 will determine the pass/fail status. When I/O0 is equal to '0', the operation passed and when equal to '1', the operation

failed. Notice where the simulation is highlighted in Figure 11, the program operation was successful.

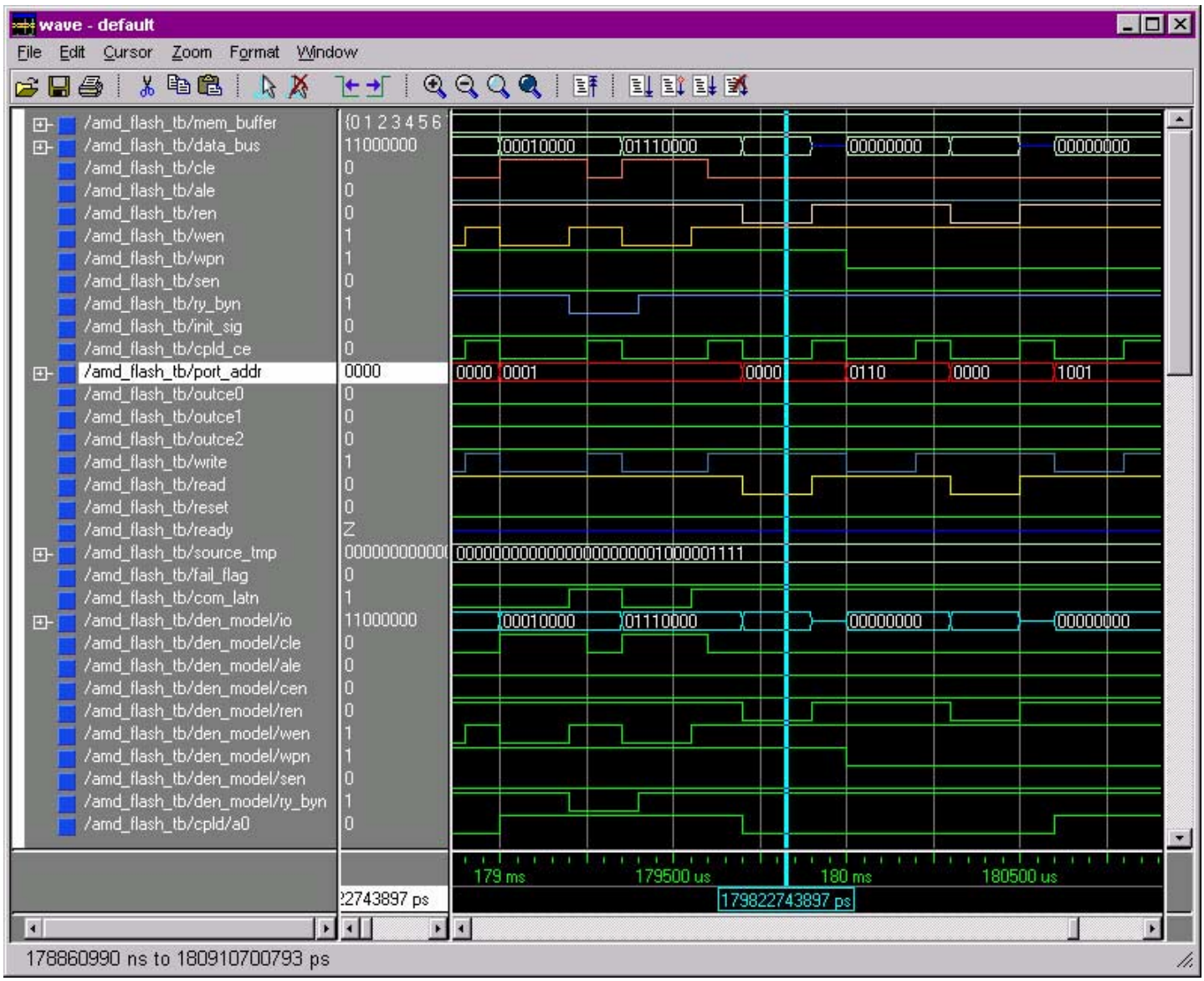

Figure 11: **Program Status: "Ready"**

Samsung Flash Memory

Testing the NAND interface with the Samsung K9F4008W0A model was performed similar to the AMD UltraNAND device. The following commands were executed with the Denali model (in ModelSim) with the test bench provided: "Frame Program" and "Status Read". To program a frame, the Frame Register must be loaded with data prior to executing the "Frame Program" command. The test bench provided loads the Frame Register with 32 bytes of data. The status of the device is checked with the "Status Read" command.

HDL Code

MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR A PARTICULAR PURPOSE, THE ADEQUACY OF THE IMPLEMENTATION, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OR REPRESENTATION THAT THE IMPLEMENTATION IS FREE FROM CLAIMS OF ANY THIRD PARTY. FURTHERMORE, XILINX IS PROVIDING THIS REFERENCE DESIGN "AS IS" AS A COURTESY TO YOU.

XAPP354 - **http://www.xilinx.com/products/xaw/coolvhdlq.htm**

Conclusion

Xilinx CoolRunner XPLA3 CPLDs are the perfect target device for interfacing with system memory devices. This NAND Flash interface can be modified to support multiple memory banks and suit any application requirements. CoolRunner CPLDs are ideal for any memory interface needs in portable and handheld devices. CoolRunner CPLDs are low power and easy to design with using the WebPOWERED software tools.

References

The web sites shown here are valid as of the publication date of this note.

1. AMD UltraNAND Flash Memory Device (**http://www.amd.com/products/nvd/overview/ultranand/ultranand.html**)
2. AMD Application Note. Simple System Interface for UltraNAND™ Flash (**http://www.amd.com/products/nvd/techdocs/22363.pdf**)
3. Samsung Flash Memory Devices (**http://samsungelectronics.com/semiconductors/Flash/Flash.htm**)
4. Denali Software, Inc. (**http://www.denalisoft.com/**)
5. eMemory (**http://www.ememory.com/**)
6. Wong, Doug. "Choosing the right Flash memory to meet your design needs". Embedded Systems Article. (**http://embedded.com/db_area/flash00/artic04.html**)

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
08/30/01	1.0	Initial Xilinx release.
09/30/02	1.1	Minor changes.