Revision	Date	Author	Comments
1A	2020-06-18	Tim S. timothystotts08@gmail.com	First publishable draft of the serial flash sector tester
2A	2020-06-19	Tim S. timothystotts08@gmail.com	Clerical updates to documentation and code.
3A	2020-07-22	Tim S. timothystotts08@gmail.com	Additional design operation and theory details.
4A	2020-07-23	Tim S. timothystotts08@gmail.com	Clerical updates to documentation.
5A	2020-07-31	Tim S. timothystotts08@gmail.com	Clerical updates to wording.

https://github.com/timothystotts/fpga-serial-mem-tester-1

Serial Flash Sector-Tester Experiment

Serial Flash Sector-Tester Experiment: Folder Structure

Serial Flash Sector Testing equivalent function of performing an erase/program/read cycle in 1/32 address steps of a 256 Mbit Serial Flash.

Project Folder	**Project Description**
SF-Tester-Design-VHDL (Vivado 2020.1)	A utility designed for a custom operation of a serial flash and displaying erase/program/read-back byte error count on both an LCD and a serial terminal. The design is completely in VHDL RTL and a simple stimulus-only visual VHDL test-bench without a soft processor. Two clock domains exist; 40. MHz and 7.37 MHz, together controlled by a MMCM; and each slower domain is controlled by a clock enable divider RTL dividing a MMCM clock by clock by clock enable pulse instead of clock division. Clock dividers are only used for the generation of an output clock that outputs at a bus port on the FPGA.
SF-Tester-Design-AXI (Vivado 2020.1 and Vitis 2020.1)	A utility designed for custom operation of a serial flash and displaying erase/program/read-back byte error count on both an LCD and a serial terminal. The design is completely in Microblaze AXI subsystem with standard Xilinx IP Integrator components, and FreeRTOS C language program executing on the Microblaze soft processor.

For the SF-Tester-Design-AXI project, the BSP of the bootloader and the BSP of the application require that the flash family parameter be set in the xilisf configuration on the BSP configure page. The default value is 1, but must be changed to 5 as the Arty-A7-100T board has a Spansion 128Mbit serial flash for booting the FPGA.

To successfully open the project, it is necessary to add the directory arty-a7-100 from the directory board_files/ to the installation directory of Vivado 2020.1 but not to the installation directory of Vitis 2020.1. For example:

```
$ which vivado
/opt/Xilinx/Vivado/2020.1/bin/vivado
```

```
$ cd ./board_files
$ sudo cp -R ./arty-a7-100 /opt/Xilinx/Vivado/2020.1/data/boards/board_files/
# (do not copy the board_files to
/opt/Xilinx/Vitis/2020.1/data/boards/board_files/)
```

Note that the Digilent Guide at https://reference.digilentinc.com/vivado/installing-vivado/2018.2 indicates that an initialization script can be executed in the user's profile to set up a path to additional board files. It is the experience of this author that the TCL initialization script provides an intermittent or non-functional detection of the board files in the user's home folder. By copying to the install directory of the tool, the board files are always found. Otherwise, the following TCL command is supposed to instruct Vivado to locate the board files:

```
set_param board.repoPaths [list "<extracted path>/Vivado/board_files"]
```

Serial Flash Sector-Tester Experiment: Methods of Operation

The purpose of the design is to boot a Digilent Inc. Arty-A7-100T (Artix-7) development board with PMOD CLS and PMOD SF3 peripheral boards, which are a 16x2 Character dot-matrix LCD display, and a 256 Mbit N25Q serial flash, respectively. The PMOD CLS and PMOD SF3 each connect to the FPGA with its own dedicated SPI bus via a higher-speed plug and jack. The PMOD CLS connects to board PMOD port JB. The PMOD SF3 connects to board PMOD port JC. The use of an extension cable makes the PMOD CLS able to connect to only one 2x6 PMOD port. See Figure 1: Arty-A7-100T Assembled with Pmod CLS, Pmod SF3.

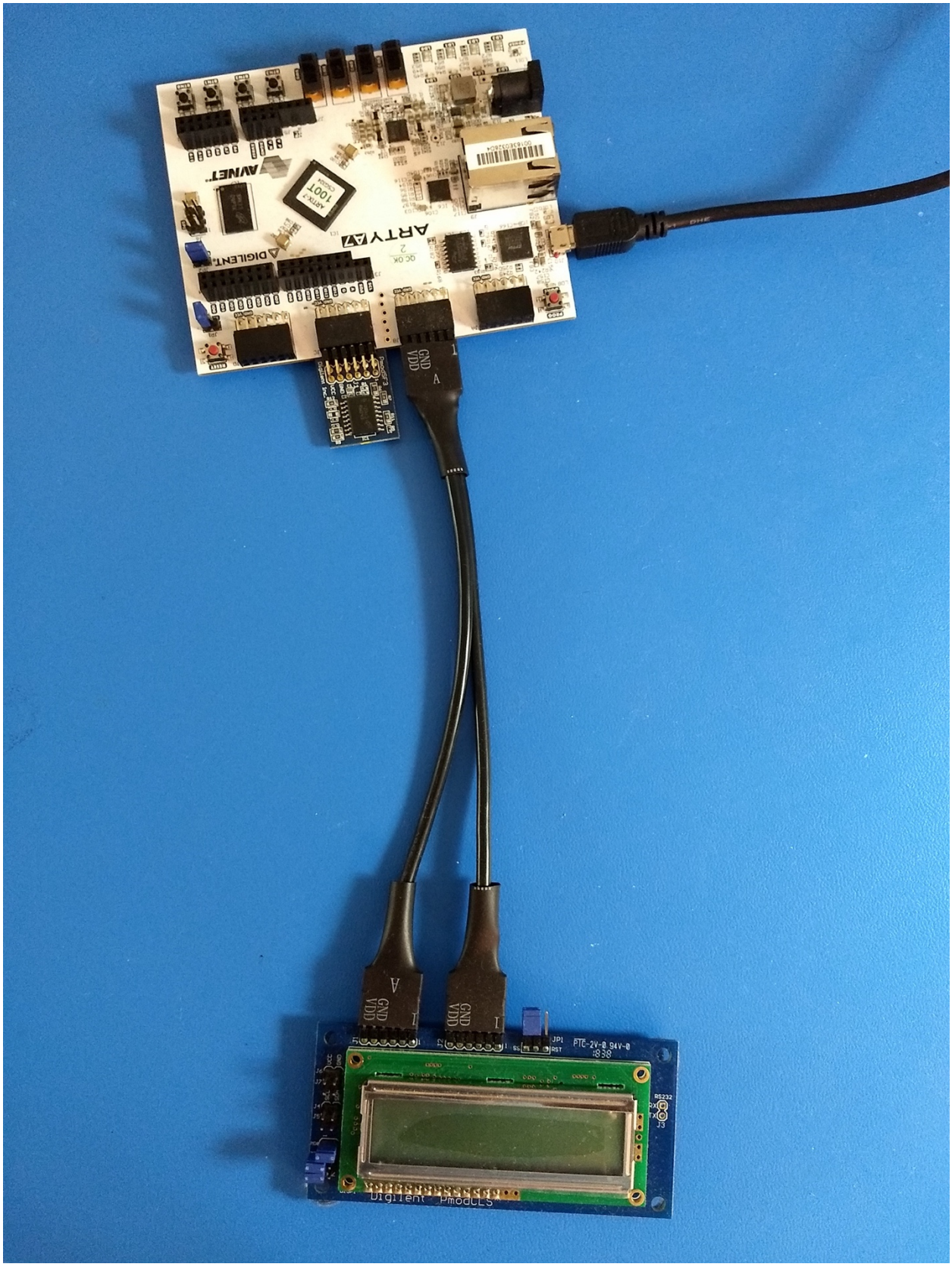

Figure 1: Arty-A7-100T Assembled with Pmod CLS, Pmod SF3

Serial Flash Sector-Tester Experiment: Method of Operation: iterative erase/program/read-back byte memory test from beginning to end of the serial flash

Serial Flash Sector-Tester Experiment: Design Operation

In the VHDL implementation, the four switches are debounced as mutually exclusive inputs, as are the four buttons. Each time a button is depressed and then released, or a switch position held as ON, a single pass of 1/32 of the memory space at the current memory index is tested for erase/program/read-back byte errors. Upon the next button depress or switch position held as ON, the subsequent 1/32 of the memory space is tested with cumulative results displayed. To test the whole memory without needing to press a button 32 times; a switch can be placed in the on position and will select the same pattern index (A, B, C, or D) as the button of the same index would have. Button 0 and Switch 0 select test pattern A. Button 1 and Switch 1 select test pattern B. Button 2 and Switch 2 select test pattern C. Button 3 and Switch 3 select test pattern D. A red color on all four LD0, LD1, LD2, LD3, indicate that the test is paused and waiting for user input; or that the test has completed. A green color on LD0, or LD1, LD2, LD3, indicates which switch or button was depressed. A white LED indicates ERASE (ERS), PROGRAM (PRO), TEST READ-BACK (TST), or DISPLAY INCREMENT (END), to show the sequential advancement of the testing of each memory area in four steps, starting with a white color on LD0, then LD1, LD2, LD3.

Note that in the AXI design, drivers downloaded from Digilent Inc. for the PMOD SF3 and PMOD CLS are used in the block design with some minimal modification and update to Xilinx Vivado and Vitis release 2020.1. The AXI implementation integrates vendor components plus adding additional C code. Note that the AXI implementation is much slower to execute each of the 1/32 cycles.

Serial Flash Sector-Tester Experiment: Design Theory

Conceptual-only FSM diagrams are also included in this portfolio in the PDF document SF3-Design-Documents/SF3-Experiment-Design-Diagrams.pdf. The final FSM diagrams show the original FSM design prior to and during coding of the first draft; and the block descriptions assist with understanding the code architecture. More complete FSM diagrams will also be included in the document, following the simpler FSM diagram page, for each major FSM designed in the HDL designs.

Note that in the IPI-BD (called AXI) Design, drivers downloaded from Digilent Inc. for the PMOD SF3 and PMOD CLS are used in the block design with some minimal modification to target the Arty-A7-100T. Both drivers target the Arty-A7 instead of the Arty. The Make files and SPI sources were updated to match usage with Xilinx Vitis 2020.1 . The AXI design integrates the vendor components plus adds additional C code. The Git repository contains a submodule that pulls from a branch of the author's fork of the Digilent Vivado-library repository on GitHub.

Coding style and choices of block design

Software design practices were used to author the VHDL sources. After the sources were drafted with a large top-level module and cohesive modules for drivers, a large self-instruction homework experiment was converted into a standalone design. The top-level HDL sources were excessively large; so modules were created to contain top-level execution-procedure FSMs, data-to-ASCII conversion combinatorial, and the addition of LED color choice control.

The led_pwm_driver.vhdl module was also refactored to infer a DSP48E1 unit without DRC errors, one unit per LED emitter. The Arty-A7 has 4 3-emitter color LEDs and 4 basic LEDs, totaling at 16 DSP48E1 being inferred to manage PWM period and duty cycle control of the eight LEDs.

Serial Flash Sector-Tester Experiment: 3rd-party references:

How To Store Your SDK Project in SPI Flash
https://reference.digilentinc.com/learn/programmable-logic/tutorials/htsspisf/start

Master XDC files for all Digilent Inc. boards, including Arty-A7-100T
https://github.com/Digilent/digilent-xdc

Digilent Inc IP library for Xilinx Vivado
https://github.com/Digilent/vivado-library/