



Electrical and Computer Engineering

ECE453

Lab 2

Basic Register Set

1. Lab 2 Overview

In Lab 2, you will develop a basic register set to learn how to develop and HDL project for the DE1-SoC.

2. ECE453 Custom Peripheral

Download the SoC base distribution from the course website. You should extract this folder to C:\shared.

You should read through sections 1-3 of the course notes that describe how to [create a custom peripheral](#). **All of the non-Verilog steps have already been completed for you!** You will need to edit the following file:

C:\shared\ece453_hdl\ip\ece453\ece453.v

In this file, you will add three registers to the Avalon register set template. The registers are described below:

Register Name	Addr	Description
DEV_ID	0x00	This is a read only register. It should return a value of 0xECE45301
GPIO_IN	0x01	This register is written by the Avalon interface. The lower 12-bits of this register are used to display a 4-digit hex code on HEX3-0 on the DE1-SoC. The upper 20 bits have no effect. A read of the register will return the current 32-bit value stored in the register.
GPIO_OUT	0x02	This register is a read only register. The bits 31-21 are unused. Bits 20-0 hold the converted 4 digit number stored in GPIO_IN. GPIO_OUT[6:0] = 7 Segment Representation of GPIO_IN[3:0] GPIO_OUT[13:7] = 7 Segment Representation of GPIO_IN[7:4] GPIO_OUT[20:14] = 7 Segment Representation of GPIO_IN[11:8]

3. Simulating Your Code

Before you attempt to build a Quartus project, make sure that you simulate your custom peripheral in ModelSim. It can take 10+ minutes to build a full SoC project in Quartus, so simulating is important if you do not want to waste time on silly syntax errors.

4. Building SoC Programming File

Use the information found [here](#) to properly connect the output of the ECE453 peripheral to the three 7-segment decoders defined by HEX2, HEX1, and HEX0. Be sure to look for ece453 in `soc_system\synthesis\soc_system.v` to determine how to connect the non-Avalon interface to external devices.

When you have finished interconnecting your peripheral, build the RBF file and start your TFTP server. This [post](#) details how to use the Tftpd64 server on your Windows PC. You will only need to read steps 4 and 5.

5. Testing Your Code from U-Boot

1. Connect the serial port of your DE1-SoC to your PC. The baud rate should be set to 115200.
2. Power on the DE1-SoC.
3. When you begin to see text messages printed out on the serial port, press the enter key to stop the Linux boot process. This will leave you at the U-Boot prompt.
4. If you have not yet done so, set the environment variables related to your Ethernet port.

```
setenv ethaddr xx:xx:xx:xx:xx:xx
setenv ipaddr yy.yy.yy.yy
setenv hostname ECE453-xx.ece.wisc.edu
setenv serverip zz.zz.zz.zz
```

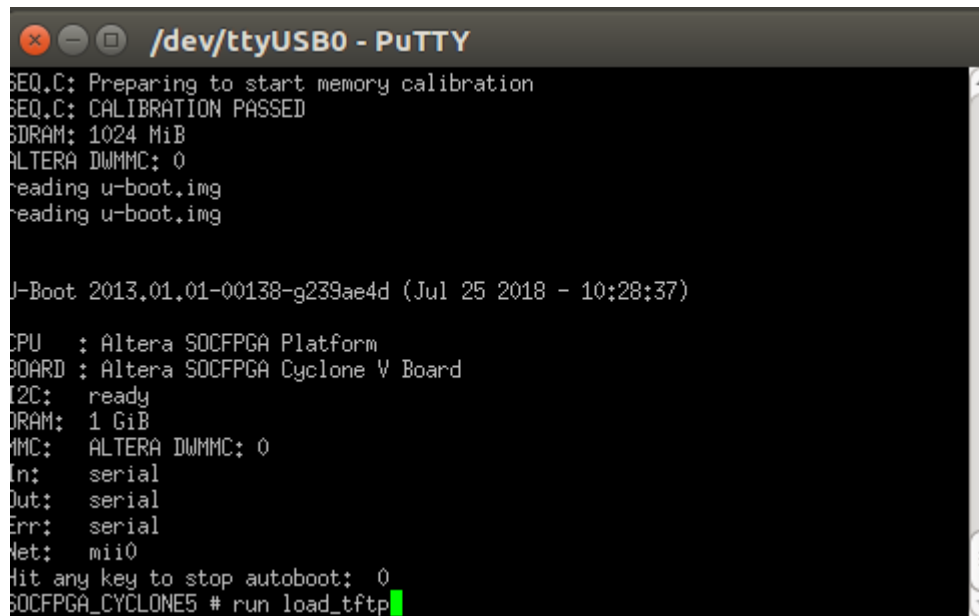
5. You can create a U-Boot environment variable that will download the RBF file from your TFTP server and program the SoC's FPGA. You can copy and paste the following lines into the U-Boot prompt.

```
setenv load_tftp tftp ${fpgadata} soc_system.rbf\; fpga load 0 ${fpgadata}
${filesize}\; run bridge_enable_handoff\;

saveenv
```

6. You can program the FPGA by running using the environment variable you created in step 5.

```
run tftp_load
```

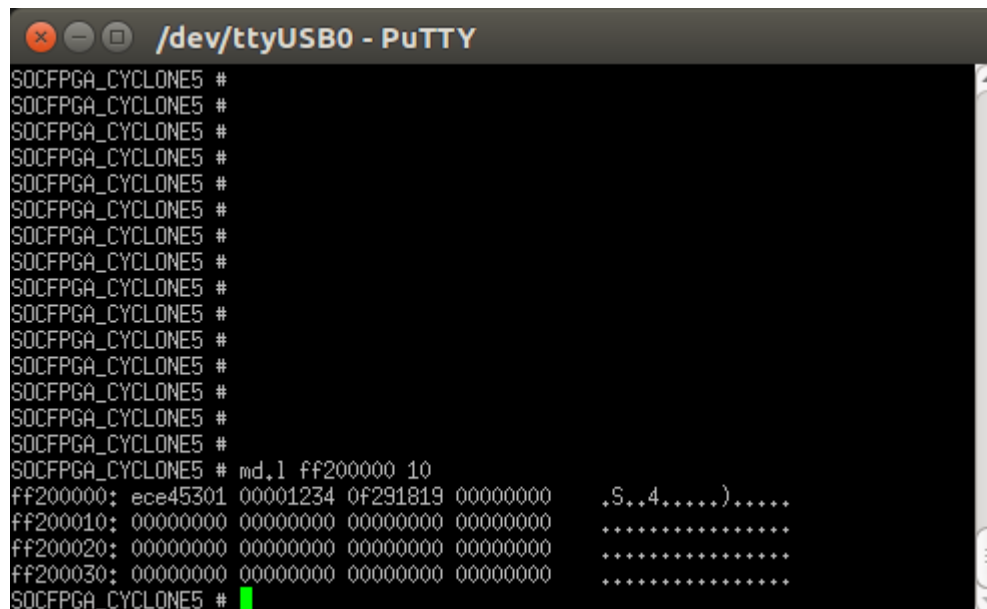


```
/dev/ttyUSB0 - PuTTY
SEQ.C: Preparing to start memory calibration
SEQ.C: CALIBRATION PASSED
SDRAM: 1024 MiB
ALTERA DWMAC: 0
reading u-boot.img
reading u-boot.img

U-Boot 2013.01.01-00138-g239ae4d (Jul 25 2018 - 10:28:37)

CPU : Altera SOCFPGA Platform
BOARD : Altera SOCFPGA Cyclone V Board
I2C: ready
DRAM: 1 GiB
MMC: ALTERA DWMAC: 0
In: serial
Out: serial
Err: serial
Net: mii0
Hit any key to stop autoboot: 0
socfpga_cyclone5 # run load_tftp
```

7. Now you can use the memory display command (md) to view an address in the memory map. In our case, the ECE453 peripheral is located at 0xff200000



```
/dev/ttyUSB0 - PuTTY
socfpga_cyclone5 #
socfpga_cyclone5 #
socfpga_cyclone5 #
socfpga_cyclone5 #
socfpga_cyclone5 #
socfpga_cyclone5 #
socfpga_cyclone5 #
socfpga_cyclone5 #
socfpga_cyclone5 #
socfpga_cyclone5 #
socfpga_cyclone5 #
socfpga_cyclone5 #
socfpga_cyclone5 #
socfpga_cyclone5 #
socfpga_cyclone5 #
socfpga_cyclone5 #
socfpga_cyclone5 # md.l ff200000 10
ff200000: ece45301 00001234 0f291819 00000000 .S..4.....).....
ff200010: 00000000 00000000 00000000 00000000 .....
ff200020: 00000000 00000000 00000000 00000000 .....
ff200030: 00000000 00000000 00000000 00000000 .....
socfpga_cyclone5 #
```

8. You can also write to the `gpio_in` register using the memory write (`mw.l`) command. After you write to this register, you should observe that the last three 7-segment LEDs display the value you wrote to the register.

[illegible]

6. What to Turn in

Folder Name	Description
Lab2_HDL.zip	<p>Your Zip file should contain the following source files:</p> <pre>ghrd_top.v ece453.v</pre> <p>You should also include a photo of the 7-segment LEDs displaying a value of 0453 on HEX3-0.</p>