Computer Assignment 4

ECGR 2181 - Fall 2009

Assignment Overview

In this assignment you will use Xilinx ISE 7.1i to Diligent Adept ExPort to create a digital circuit and program it to the Nexys2 Spartan 3E FPGA. The digital circuit will store the values of the 8 sliding switches (up = '1' / down = '0') and display them to the 8 LEDs (on = '1' / off = '0') when the "Set" push button is pressed. While the "Flip" push button is pressed the output of the LEDs should be inverted (if LED0 was previously ON it should now be OFF). When the "Flip" push button is released the output of the LEDs should return back to their un-inverted (original) values. When the "Reset" push button is pressed the LEDs should all be turned OFF.

One additional VHDL File (debounce.vhd) has been provided which is available on the class website under the "Computer Labs" section. Demo the final circuit on the FPGA to one of the TAs during their Office Hours (in Woodward 200). Since there are a limited number of FPGAs we request that you only check out an FPGA right before you need to demo the working circuit. Use ModelSim to test the functionality of the circuit (saving you lots of hardware debugging time) then once the circuit works check out the FPGA from the ECE Shop. **START EARLY! There are fewer boards available than groups, so if you wait until the due date to start you might not get a board to test and demo with.**

You are allowed to work in groups of 2 under the following conditions:

1. Both partners are in the same class section.
2. You must work with someone new. You cannot work with the same partner as CA3.
3. Both partners must work together and be able to answer any questions asked during the demo.

In addition you will need to upload a **single PDF** of the listed files along with printing out and handing in the Grade Rubric. Print each page to a PDF file and use Adobe Acrobat Pro to combine the pages to a single PDF. Name the PDF <lastname_lastname_cad4>.pdf (schmidt_conrad_cad4.pdf). Upload the PDF to Moodle under Computer Assignment 4.

Computer Assignment 4 is due Wednesday, December 2nd by 3 PM.

Getting Started

1. Create a new directory to store all of the Computer Assignment Files
    a. Make sure the directory does not contain any spaces!
2. Open **Xilinx ISE 7.1i** and Create a New Project called: **cad_4**
3. Create it for the **Nexys2 Board** (Spartan 3E, xc3s500e, fg320, -4)
4. Add 1 New Source File of type Schematic: **cad_4_top_sch**
    a. This will be the top level design for your design.
5. Under Add Existing Sources add the debounce.vhd VHDL from the Class Website making sure to check the "Copy to Project" check box. Click Next and select "VHDL Design File" when prompted.
6. Click Next and Finish to exit the New Project Wizard.

7. Add the following new files to the design:
    a. d_latch_sch – Schematic Type
    b. d_flip_flop_sch – Schematic Type
    c. register_1_bit_vhdl – VHDL Module Type
        i. d : in std_logic
        ii. set : in std_logic
        iii. rst : in std_logic
        iv. flp : in std_logic
        v. clk : in std_logic
        vi. q : out std_logic
    d. register_8_bit_sch – Schematic Type

d_latch_sch

1. Create the Level Sensitive D Latch from Gates (refer to the notes if necessary).
2. Generate the Schematic Symbol under the Process View Tab's Design Utilities Menu.

d_flip_flop_sch

1. Create the Rising Edge D Flip-Flop from the Level Sensitive D Latch.
2. Generate the Schematic Symbol.

register_1_bit_vhdl

1. The 1-bit register will consist of 3 parts.
    i. Select the correct input to the D Flip-Flop
    ii. Store the data into the D Flip-Flop
    iii. Select the output of the register from either the D Flip-Flop's **q** or **q_n**
2. The 1-bit register should have the following functionality:
    a. When **rst** = '1' the D Flip-Flop input **d** should be '0'
    b. When **set** = '1' **d** should be stored into the D Flip-Flop's **d** input
    c. When **set** and **rst** = '0' the register's output should be the D Flip-Flop's **q** output
    d. When **flp** = '1' the register's output **q** should be the D Flip-Flop Output **dff_qn**

    Notice the difference between the 1-bit register's output **q** and the D Flip-Flop's **q**
3. Add the following signals between the "architecture" and "begin" of the VHDL file:

```
architecture Behavioral of register_1_bit_vhdl is
    -- D Flip-Flop's Q' Output Signal (used by 2:1 Mux)
    signal dff_q_n     : std_logic;
    -- D Flip-Flop's Q Output (used by both Muxes)
    signal dff_q       : std_logic;
    -- 4:1 Mux Select Signal (notice it is 2 bits)
    signal mux4_select : std_logic_vector(1 downto 0);
    -- 4:1 Mux Output Signal (connects to what component's input?)
    signal mux4_out    : std_logic;
    -- 2:1 Mux Select Signal (notice it is only 1 bit)
    signal mux2_select : std_logic;

    -- D Flip-Flop Schematic Component created Already
    component d_flip_flop_sch
    port(q_n :  out std_logic;
         q   :  out std_logic;
         d   :  in  std_logic;
         clk :  in  std_logic);
    end component;

begin
```

4. Set the **mux4_select** and **mux2_select** signals based on the previously described functionality.

(Hint: The 4:1 Mux depends on “set” and “rst” while the 2:1 Mux depends on “flp”)

5. Instead of creating a separate 4:1 and 2:1 1-bit Multiplexers you can just add each process directly into the register_1_bit_vhdl between the “begin” and “end Behavioral”
6. Create a New Process called: **mux4_process** which will have the following:
    a. Sensitivity List = dff_q, d, mux4_select
    b. A Case Statement to “select” the D Flip-Flop’s input based on “set” and “rst” signals.
7. Create a New Process called: **mux2_process** which will have the following:
    a. Sensitivity List = dff_q, dff_q_n, mux2_select
    b. A Case Statement to “select” the output of the register’s **q** based on the “flp” signal
8. Add the D Flip-Flop Instance Code:
    (The line numbers might not match exactly but this code should be after the two muxes)

```
84      d_ff: d_flip_flop_sch
85      port map(
86         q_n => dff_q_n,
87         q   => dff_q,
88         d   => mux4_out,
89         clk => clk
90      );
```

9. ModelSim – It is **HIGHLY** recommended that you test your 1-bit register with ModelSim. This will save you headaches and debugging time. You will want to make sure to assert the “rst” signal for at least the first 2 clock cycles so the D Flip-Flop can be initialized to ‘0’. Then you can test setting and flipping the bit.
10. Create the Schematic Symbol for the register_1_bit_vhdl to be used next.

register_8_bit_sch

1. Design an 8-bit register based on the register_1_bit_vhdl component. Use buses for the Input **d(7:0)** and output **q(7:0)**
2. Create the Schematic Symbol for the register_8_bit_sch
3. ModelSim – As with the 1-bit register it is **HIGHLY** recommended to test with ModelSim.

cad_4_top_sch

1. The Top Level Component will have the following inputs and outputs:
    a. Inputs: **set_sw**, **rst_sw, flp_sw, clk, d(7:0)**
        i. **d** will be the 8 input slide switches on the Nexys2 Spartan 3E board
    b. Outputs: **q(7:0)**
        i. **q** will be the 8 LEDs above the slide switches on the Nexys2 Spartan 3E board
2. Add the **register_8_bit_sch** to the schematic
3. Create the Schematic Symbol for the debounce-behavorial (debounce.vhd) component
4. Add 3 debounce components and connect the 3 push button inputs (**set_sw**, **rst_sw**, **flp_sw)** to each debounce’s **sw** input. Connect the **clk** to the debounce’s **clk** input. Connect the **sig** output from each of the three debouncer’s to the correct input on the register_8_bit_sch. Leave **sglPulse** output from the debouncers unconnected.
5. Under the Process View Expand User Constraints and double click on Edit Constraints (Text)

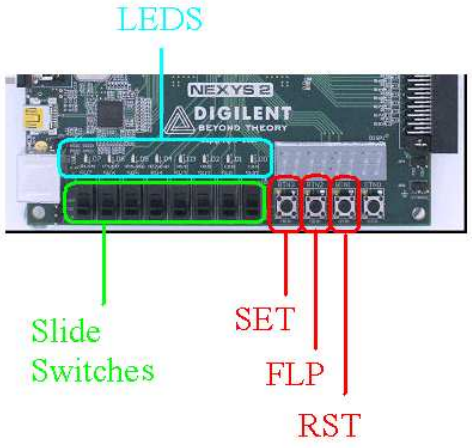
6. Copy and Paste the following constraints. This is connecting the physical pins for the LEDs, Switches and Clock on the FPGA to the top level component's input and outputs.

```
NET "clk"  LOC = "b8" ;
NET "d<0>"  LOC = "G18" ;
NET "d<1>"  LOC = "H18" ;
NET "d<2>"  LOC = "K18" ;
NET "d<3>"  LOC = "K17" ;
NET "d<4>"  LOC = "L14" ;
NET "d<5>"  LOC = "L13" ;
NET "d<6>"  LOC = "N17" ;
NET "d<7>"  LOC = "R17" ;
NET "flip_sw"  LOC = "E18" ;
NET "q<0>"  LOC = "J14" ;
NET "q<1>"  LOC = "J15" ;
NET "q<2>"  LOC = "K15" ;
NET "q<3>"  LOC = "K14" ;
NET "q<4>"  LOC = "E17" ;
NET "q<5>"  LOC = "P15" ;
NET "q<6>"  LOC = "F4" ;
NET "q<7>"  LOC = "R4" ;
NET "rst_sw"  LOC = "D18" ;
NET "set_sw"  LOC = "H13" ;
```

7. Double click "Synthesis – XST" - Synthesize the top level component and check for errors.
8. Double click "Implement Design" – The Tools determine where to put the logic on the FPGA
9. Double click "Generate Programming File" – Creates the bitstream needed to program the FPGA with for your design. The generated .bit file will be used by ExPort in the next step.

Programming the FPGA – Downloading the Design to the FPGA

1. Open Diligent Adept ExPort from the Start Menu (Refer to the Computer Assignment 0 Tutorial)
2. Plug in the Nexys2 Spartan 3E FPGA board into the USB port of the Computer (you will need to check out this board from the ECE Shop at this point).
3. Turn on the board if it isn't already on (there is a slide switch in upper left corner of the board)
4. In the program ExPort click the button "Initialize Chain" (PC identifies which FPGA is present)
5. Browse to your **.bit** file generated in the previous Step 9
6. Click "Yes" to the question of using "CCLK" instead of "JTAG CLK"
7. Click the Check Box next to the XCF04S ROM (Do not browse to find it)
8. Click the "Program Chain" Button to program the FPGA and Test Circuit



Demo

1. When you are ready to show the TA your code is correctly functioning on the FPGA find the TA to take the demo. The TA will test the design for various input and outputs.
2. The TA will ask questions regarding the Computer Assignment prior to accepting the grading rubric and signing off on it.

Required Electronic Submission (uploaded to Moodle)

Print the Grading Rubric (the last page of this hangout) and write both names and section number at the top. Then print the following schematics and VHDL files as individual PDFs. Combine the PDFs into a single PDF with both your last names as the filename (I.E. schmidt_conrad_cad4.pdf). Upload the single PDF (one upload per group) to Moodle. Finally, find one of the TAs **during their Office Hours** and demo.

1. Schematic of d_latch_sch
2. Schematic of d_flip_flop_sch
3. VHDL of register_1_bit_vhdl
4. Schematic of register_8_bit_sch
5. Schematic of cad_4_top_sch

Notes

If you run into a License Error during Simulation save all files and close Xilinx ISE 7.1i. The open Xilinx ISE 7.1i from the Start Menu:

Start -> Mosaic XP -> Engineering -> Electrical -> Xilinx Webpack -> Xilinx ISE 7.1i

If you run into a ModelSim "read only" error on your Test Bench VHW file:

a. To avoid getting any errors with ModelSim in "read mode" you MUST create a new project directory WITHOUT any spaces! Make sure the path to the directory also does not have any spaces. If you are not sure where to save this project to, create the following directory on your desktop:
    i. ecgr2181
    ii. Then within this directory add a new directory for each project
    iii. The Path to this directory will now be: **U:\pc\win_data\Desktop\ecgr2181**
b. Make sure you Open your project from the U drive (not from the AFS network). When opening your existing project in Xilinx ISE 7.1i select My Computer -> U:\ and find the .ise file where your project is located.

Computer Assignment #4 - Grading Rubric

Student 1 Name: _____ Section Number: _____

Student 2 Name: _____

Uploaded correctly functioning schematics and VHDL to Moddle:

- Schematic of d_latch_sch _____ / 2 Points
- Schematic of d_flip_flop_sch _____ / 3 Points
- VHDL of register_1_bit_vhdl _____ / 5 Points
- Schematic of register_8_bit_sch _____ / 3 Points
- Schematic of cad_4_top_sch _____ / 2 Points

Demo on the FPGA:

- Demo of Circuit on FPGA _____ / 15 Points

Total _____ **/ 30 Points**