Cpr E 281 FINAL PROJECT
ELECTRICAL AND COMPUTER ENGINEERING
IOWA STATE UNIVERSITY

FINAL Project

Objectives

The main objective of the final project is to teach you how to put together all of the class material that you have learned so far in order to program the Altera DE2 board to carry out an independent activity. You will have to design a circuit to realize one digital machine that accepts input from switches and outputs to LEDs and 7-segment displays.

Project Selection

Feel free to choose any one of the following three projects. They are all of approximately the same difficulty. Your grade will not depend on the project that you pick, as long as you implement the entire project.

If you don't like any of the provided projects, then you also have the option of specifying your own project. If there was something that you always wanted to do in digital design, now is your chance. However, your proposed project MUST be of appropriate complexity as determined by your lab TA. The minimum constraints are that y*our project MUST: 1) include a state machine, 2) include a register file, 3) a combinational logic, and 4) be demonstrated on the ALTERA boards provided in the lab.*

Notes on the Three Provided Projects

The three projects descriptions outline all necessary details for the projects. If you choose to implement one of these projects, **you must follow** all details listed in the project description. Your implementation, at a minimum, **should satisfy** all implementation features and constraints as specified in this document. This means that if you do not implement a certain feature or reduce the complexity of the project you will lose points as indicated in the grading **RUBRIC**. You are allowed to work beyond the basic functionality as long as the implementation improves upon the specified design. It would be helpful to discuss any such changes with your TA.

Think about the final project as a really long homework. You may have to go to the lab outside of your regularly scheduled lab time in order to complete it. To get a grade for this assignment you need to present your final project to your TA during dead week (in your regular lab time). No presentation, no grade!

Also, there will be ZERO credit for any design step in the grading rubric that does not work or produces the wrong output or is not of the given specifications. All components of your design MUST WORK and produce the output as described in the specs to receive credit.

Project #1: Tiny Encryption Algorithm Encoder/Decoder

See the grading RUBRIC below before starting with the design.

Implement an Encoder/Decoder for the basic tiny encryption algorithm (TEA-- https://en.wikipedia.org/wiki/Tiny_Encryption_Algorithm). TEA takes two 32-bit integers (V0 and V1 -- really a 64-bit value) and four 32-bit keys (K0, K1, K2, and K3 -- really a 128-bit key) and encrypts or decrypts V0 and V1 in a relatively simple and efficient manner.

This circuit should have a control FSM and a datapath. The datapath should contain a register file, an arithmetic and logical unit (ALU), and some step counters. The register file should have one write port and two read ports and contain ten 32-bit registers (V0, V1, K0, K1, K2, K3, sum, tmp0, tmp1, and tmp2) and a 32-bit ``magic constant'' (0x9e3779b9). The ALU should take two 32-bit operands and support all the necessary operations to perform the mixing step (i.e., several types of shifting, bitwise or, bitwise xor, addition, and subtraction). Use as many counters as necessary to aide in control reading in input values and writing out the output values.

Operation:

0. Load the key: If toggle switch 8 is asserted, you should be able to load K0-K3, one byte at a time by setting switches 0-7 and pushing pushbutton 0.

1. Load the values for encoding/decoding: If toggle switch 8 is not asserted, you should be able to load V0 and V1, one byte at a time by setting switches 0-7 and pushing pushbutton 0.

2. Encoding/decoding: Once the last byte of V1 is entered, encoding or decoding begins (toggle switch 9 determines whether the circuit encodes values or decodes them).

3. Display: Once encoding/decoding is complete, V0 and V1 should be displayed on the 7-segment displays one byte at a time with pushbutton 0 advancing the byte.

RUBRIC

Each of the following steps must be in its **own design file**. Demonstrate each step individually to receive credit.

a. Design and demonstrate the two-port read and one-port write register file works by writing and reading from the register file, including reading in the same cycle. **(20 points)**

b. Design and demonstrate the ALU circuit. **(20 points)**

c. Design and demonstrate that the state machine works. **(20 points)**

d. Put together all the individual components in Parts a, b, and c and demonstrate that the complete TEA encoding and decoding works.
(20 points)

e. Final Report **– (20 points)**

Project #2: 8-bit Booth's Multiplier

See the grading RUBRIC below before starting with the design.

As you learned earlier in the semester, multiplication is effectively repeated shifting and addition. This understanding can be used to reduce the area of a hardware multiplier by implementing a sequential multiplication circuit. Unfortunately, this means that for each bit of the multiplier there is both an add and a shift, even for cases like *0 where no adds are necessary. Therefore, you are tasked with designing an 8-bit sequential multiplier based on Booth's algorithm (https://en.wikipedia.org/wiki/Booth%27s_multiplication_algorithm). Your Booth's algorithm multiplier should have an FSM for controlling the algorithm and a datapath that includes a register file, a modified adder-subtractor circuit (needs selective sign extension), and some single flip-flops to represent P[-1]/carries/signs.

The register file has two read ports, a single write port, and contains just four 8-bit registers (multiplicand--m, lower result--P[7:0], upper result--P[15:8], and count).

Behavior:

Step 1: Initialization

* Load the multiplicand from toggle switches 0-7.
* Load the multiplier into the lower result register from toggle switches 8-15.
* P[-1] is initialized to 0.

Step 2: Multiplication

* Read P lower and check the value of p[0:-1]:

  If 01 add a sign-extension of m to P (this is a multi-step process since P is a 16-bit value).

  If 10 add a sign-extension of -m to P (this is a multi-step process since P is a 16-bit value).

  If 00 or 11 don't add or subtract.

* Right shift P (use an arithmetic shift where P's sign doesn't change).
* Repeat these steps until they have been done 8 times.

Step 3: Display result

* Display the result (i.e., P[15:0]) on the 7-segment display using pushbutton 0 to toggle between the upper and lower half.

RUBRIC
Each of the following steps must be in its **own design file**. Demonstrate each step individually to receive credit.

a. Design and demonstrate the two-port read and one-port write register file works by writing and reading from the register file, including reading in the same cycle. **(20 points)**

b. Design and demonstrate the modified adder/subtractor circuit. **(20 points)**

c. Design and demonstrate that the state machine works. **(20 points)**

d. Put together all the individual components in Parts a, b, and c and demonstrate that the complete multiplier works. **(20 points)**

e. Final Report. **(20 points)**

Project #3: Key Pad lock

See the grading RUBRIC below before starting with the design.

Create a functional door lock that has the following Features/Functionality:

1. Unlock door after a 4 digit password has been entered using push buttons, e.g., the push buttons pressed one at a time and in this order 1,2,3, and 4 would result in a code of 1234.
2. Lock indicates the current entered code on 4 7-seg displays. Non-entered digits should not display any number.
3. Lock checks code after the 4th button has been pressed and clears current entered code
4. Lock has a clear code button to allow user to clear the current entered code
5. Code will be stored in an 8 bit register and can be modified after the correct code is entered
6. Lock keeps track of incorrect attempts
7. Lock will lock until hard reset if number of incorrect attempts exceeds 10
8. Lock locks after lock door switch is flipped
9. Lock indicates when it is locked, unlocked, locked until hard reset, and entering in a new code.
10. After hard reset, the code is 1111, and number of incorrect attempts is 0.

You will need registers to store the following:

- For Code that opens the Lock
- The code entered by the user
- Number of incorrect attempts

The boards in the lab have only 4 push buttons. This limits the valid digits in the code to 1, 2, 3, and 4. That is, a code of length 4, where each of the four digits can be in the range 1-4. This still gives you 4x4x4x4=256 possible lock combinations.

RUBRIC
Each of the following steps must be in its **own design file**. Demonstrate each step individually to receive credit.

a. Enter, display, and clear entered code. **(20 points)**
b. Demonstrate the ability to edit and store new unlocking code (this step is testing that you can make a register file and can edit/access its contents, not necessarily that the door unlock works). **(20 points)**
c. Demonstrate a functional door lock with code. **(20 points)**
d. Put together all of the individual components in Parts a,b, and c and demonstrate that the door will remain locked until a hard reset after 5 incorrect attempts. **(20 points)**
e. Final Report. **(20 points)**

Project #4: Specify Your Own Project

See the grading RUBRIC below before starting with the design.

Propose a project of your own design. Once again, the minimum constraints are that your project MUST: 1) include a state machine, 2) include a register file, 3) a combinational logic, and 4) be demonstrated on the ALTERA boards provided in the lab.

If you choose this option, you **MUST** discuss your intended design with your lab TA to see if the project would be appropriate for this class. This discussion can be over e-mail. Please contact your lab TA.

If the lab TA or Instructor evaluates the project and recommends that it is not of appropriate difficulty, then the project will be denied and you'll have to modify it or propose a new one.

RUBRIC
Each of the following steps must be in its **own design file**. Demonstrate each step individually to receive credit.

a. Design and demonstrate the state machine for your design **(20 points)**

b. Design and demonstrate the register file for your design **(20 points)**

c. Design and demonstrate the combinational circuit. **(20 points)**

d. Put together all of the individual components in Parts a,b, and c and demonstrate the complete project. **(20 points)**

e. Final Report. **(20 points)**