Name: _____ ID# _____

Date Submitted: _____ Lab Section # _____

CSE 2441 – Introduction to Digital Logic Fall Semester 2020

Lab Number 10 – TRISC Registers and Counters

(Due – November 9, 2020, 11:59 PM)

This is a DE1 FPGA Lab.

CSE 2441 **LABORATORY ASSIGNMENT 10** **FALL 2020**

TRISC REGISTERS AND COUNTERS

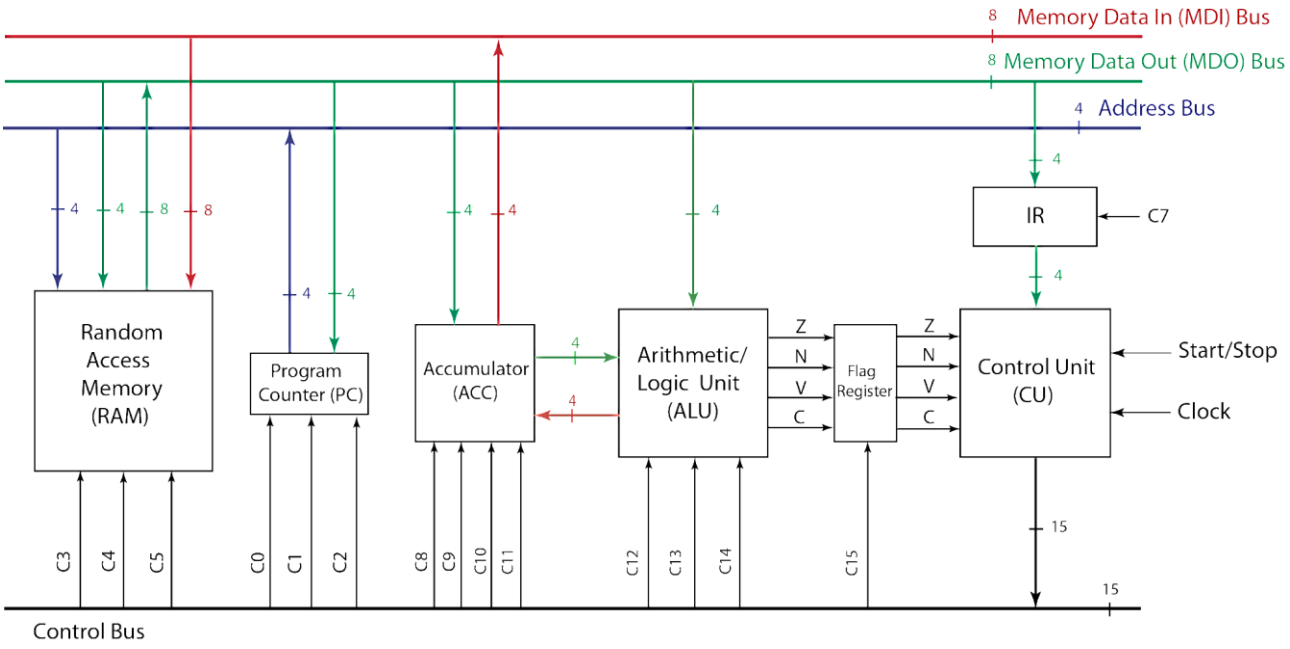
(100 POINTS)

PURPOSE/OUTCOMES

To design, implement on the DE1, and test registers and counters needed for the Term Design Project. Once you complete this assignment, you will have demonstrated an ability to design components found in stored-program digital computers.

BACKGROUND

The CSE 2441 term design project involves the design, implementation, testing, and demonstration of the Tiny RISC (TRISC) processor. TRISC has the hardware structure and organization shown below. See the Appendix for control signal definitions.



In this lab (#10), you will design, implement, and test the Instruction Register (IR), Program Counter (PC), and Accumulator (ACC). In Lab #11, you will study the operation and interfacing of Random Access Memory (RAM). In Lab #12, you will design, implement, and test the Control Unit (CU). You previously designed the Arithmetic/Logic Unit (ALU) in Lab #7.

The Term Project will consist of integrating the previously designed and tested components and demonstrating that your processor can execute a simple program. More details on the term project will be provided later.

DESIGN ASSIGNMENT (50 points)

1. Design the Instruction Register (IR) using the parallel-in/parallel-out (PIPO) register from Lab #6. The IR stores the four-bit opcode of the instruction to be executed on the processor. The IR is loaded from the Memory Data Out (MDO) bus upon receipt of control signal *C7*. Create a symbol file for your IR.

2. Design the Program Counter (PC) using the binary up-counter from Lab #6. The PC keeps track of the address of the next instruction to be fetched from RAM. The PC is loaded from the MDO bus on receipt of control signal *C1*, cleared on *C0*, and incremented on *C2*. Create a symbol file for your PC.

3. Design the Accumulator (ACC) using the binary up-counter from Lab #6 and a quad two-to-one multiplexer. The ACC holds one operand for certain arithmetic/logic operations such as ADD, SUBTRACT, AND, XOR and is also used to store the result of such operations. As such, the ACC is loaded from either the MDO or the ALU. Control signal *C10* is used to select the input source. The ACC is loaded on receipt of *C11*, cleared on *C8*, and incremented on *C9*. Create a symbol file for your ACC.

LAB ASSIGNMENT (50 points)

1. Implement your IR design on the DE1 using the following pin assignments.

   Data inputs – SW3, SW2, SW1, SW0
   Data outputs – LEDR3, LEDR2, LEDR1, LEDR0
   C7 – Key1

   Demonstrate that your IR can load input patterns 0000, 0101, 1010 and 1111.

2. Implement your PC on the DE1 using the following pin assignments.

   Data inputs – SW3, SW2, SW1, SW0
   Data outputs – LEDR3, LEDR2, LEDR1, LEDR0
   C0 – Key0
   C1 – Key1
   C2 – Key2

   Demonstrate that your PC can load input patterns 0000, 0101, 1010, and 1111.
   Demonstrate that you can clear your PC with C0.
   Demonstrate that you can increment your PC from 0000 to 1111 using C2.

3. Implement your ACC on the DE1 using the following pin assignments.

   MDO data inputs – SW3, SW2, SW1, SW0
   ALU data inputs – SW7, SW6, SW5, SW4
   Data outputs – LEDR3, LEDR2, LEDR1, LEDR0
   C8 – Key0
   C9 – Key1
   C10 – Key2
   C11 – Key3

   Demonstrate that you can clear your ACC with C8.
   Demonstrate that your ACC can count from 0000 to 1111.
   Demonstrate that you can load your ACC with 0101 from the MDO inputs.
   Demonstrate that you can load your ACC with 1010 from the ALU inputs.

REPORT REQUIREMENT

1. Cover sheet (as shown on this assignment)
2. Lab purpose
3. Circuit diagrams of your realizations (screen shots)
4. Symbol file diagrams (screen shots)
5. Test results for the IR (*)
6. Test results for the PC (*)
7. Test results for the ACC (*)

* Document with photos or videos

APPENDIX

Control Signal Definitions (Active High)

C0 – *Clear* Program Counter (PC)

C1 – *Load* PC

C2 – *Increment* PC

C3 – *Select* Memory Address Register (MAR) source (C3=0: PC, C3=1: MDO)

C4 – *Execute* RAM READ/WRITE cycle

C5 – *Enable* RAM WRITE cycle

C7 – *Load* Instruction Regiser (IR)

C8 – *Clear* Accumulator (ACC)

C9 – *Increment* ACC

C10 – *Select* ACC source (C10=0: ALU, C10=1:MDO)

C11 – *Load* ACC

C12,C13 – Arithmetic Logic Unit (ALU) Function Code (00: ADD, 10: SUB, 01: AND, 11: XOR)

C14 – *Load* Buffer Register (BR)

C15 – *Load* Flag Register (FR)