Fall 2019 CSE 4117 HW 1

A group consists of exactly two people (One person groups are not allowed)
Due date is November 1, 2019. Demo date will be announced later.
Note: The two questions are completely independent of each other.

FIRST QUESTION

In this question, you will construct the FROG (The third CPU designed in class) in Logisim and Verilog. In addition to the given instructions of Frog (ldi, add, sub, inc, dec, or, and, xor, mov) you will implement 5 immediate arithmetic logic instructions (addi, subi, ori, andi, xori).

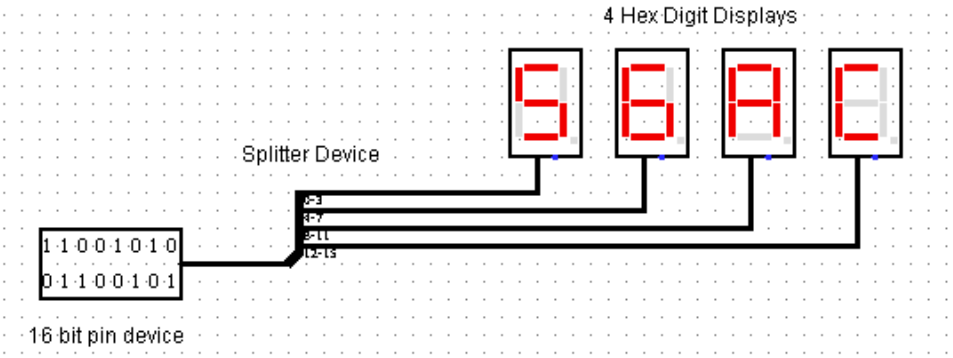
addi r1,r2,imm	r1=r2 + imm
subi r1,r2,imm	r1=r2 - imm
andi r1,r2,imm	r1=r2 & imm
ori r1,r2,imm	r1=r2 | imm
xori r1,r2,imm	r1=r2 ^ imm

where imm is a 16-bit integer and r1and r2 are any two 16-bit registers.

The tasks to be accomplished are enumerated below:

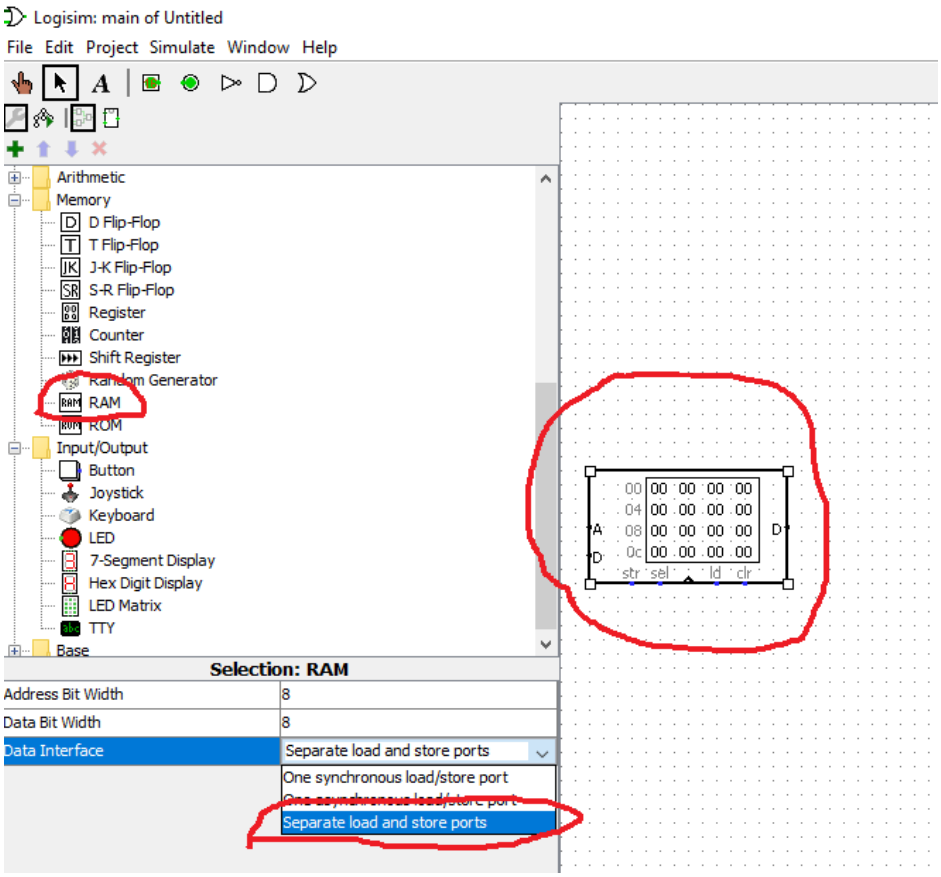
Logisim Part:

1. Build the frog using Logisim with the added immediate instructions defined above.
2. Attach 4 7-segment displays to the output of Register 0. In Logisim there are two different kinds of 7-segment displays: "7-segment display" and "Hex Digit display". Use "Hex Digit Display" (it is in input/output menu on the left side of the workspace).



In the figure above 16-bit data 0x56AC is displayed in 4 Hex Digit Displays with the aid of a splitter device (it is in the wiring menu on the left side of the workspace), which divides 16 bits into 4 4-bit chunks. Note that hex digit displays have 4 bit inputs and display hexadecimal data directly and hence they are much easier to use than the regular 7-segment displays.

3. The size of the RAM chip is to be decided by you.



Use two ported memory as shown above. Note that in the above figure the data and address bus width are 8-bit. Yours must be different. Also as you won't write into the memory (Recall that frog does not have the capability to write into the memory, it only uses memory to read instructions) you should leave the data input of the memory open (or, better, you can just connect 0 to data input).

4. Complete the assembler given in the website to work with the above design.
   https://microprocessors.gitbooks.io/deneme/content/frog1.html
   Note that the assembler code is incomplete. **You are also required to add the new immediate instructions to the assembler.**

5. Write a program that will calculate the expression given below and put the result into register 0.

   R0 = [(((107|24)--)+!(27&45))++]^!(12-95)

   You should use some of the immediate instructions when evaluating above expression.

NOTE THAT

In Logisim, you can use

- SR flip flops
- Logical Gates
- Encoders and decoders
- Multiplexers and demultiplexers
- Adders
- Memory (ROM and RAM)
- Input and output pin devices

and nothing else, ie, you must construct everything else out of these.

Verilog Part:

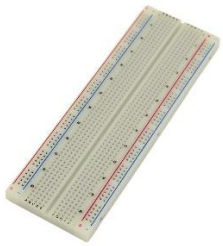
Do the same in Verilog (including the new immediate instructions) and realize it in Altera DE0-nano FPGA kit.

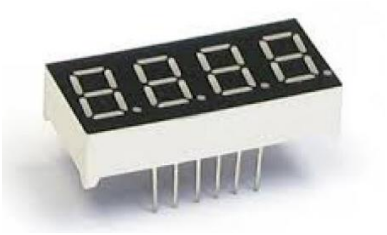
In particular;

- You should understand the code for frog given in https://microprocessors.gitbooks.io/deneme/content/frog1.html
- You should understand the 7-Segment driver code which drives 4 7-Segment displays given in the same location
- You should write a top module which connects the 7-Segment driver code into the register 0 of frog. i.e. the 7-Segment registers must automatically display the hexadecimal number that is in register 0.
- You should debug and compile the code constructed above.
- You should use push button of the DE0 Nano Card as the clock of the frog, i.e. your code must be executed as you press the push button.
- You should run your 7-segment driver code in 50 Mhz clock of DE0 Nano Card.

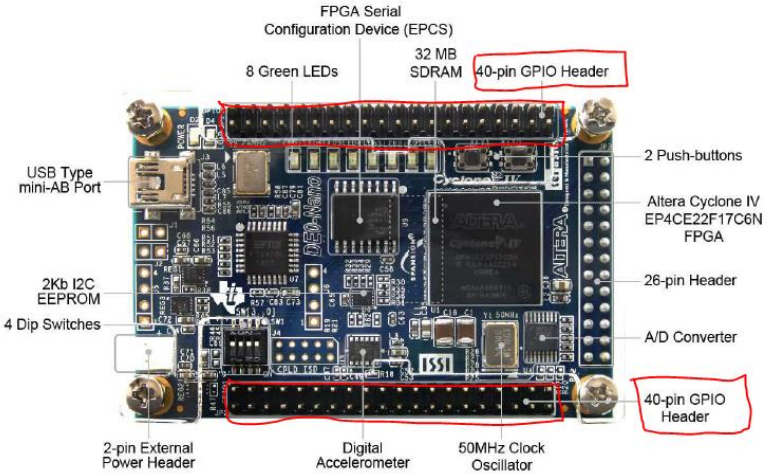
To achieve these tasks, you need to buy;

- A breadboard,
- One jumper cable (Female to male),
- A 4-digit seven-segment display (Common Cathode).





You have to do pin assignments to connect the pins in the GPIO ports of the FPGA (shown circled in red in figure below) to the 7-Segment displays placed on breadboard via the jumper cable as described in the class.



SECOND QUESTION

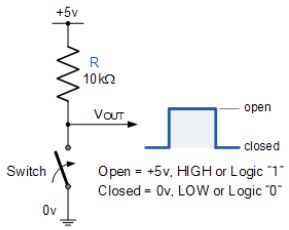
In this question, you will construct a 16-bit display unit in Verilog and realize it in Altera DE0-nano FPGA kit. You will use the same hardware as above but you will add

- 1x 4 bits dip-switches,
- 4x 1K resistors,

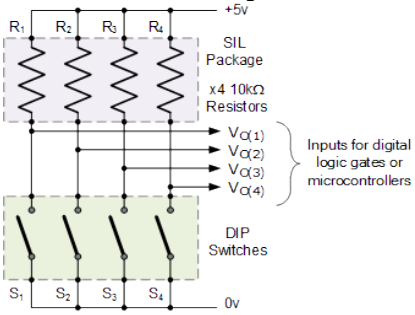
You will construct a Verilog code which will display the student numbers of both members of the group in a concatenated way. i.e. if the numbers 123456789 and 444444444, you will display 123456789444444444. This number will be displayed in 4-digit 7-Segment display in a sliding manner. The manner of the slide will be determined by the switches of the dip-switch.

- Switch 1 controls whether the speed is normal or double.
- Switch 2 controls whether we are operating on a rotate or swing mode.
- Switch 3 controls the direction of the rotation, i.e., whether it is from left to right or right to left. Switch 3 is not functional if we are on swing mode.
- Switch 4 controls whether the output blinks for a period of ¼ seconds (approximately) or not.

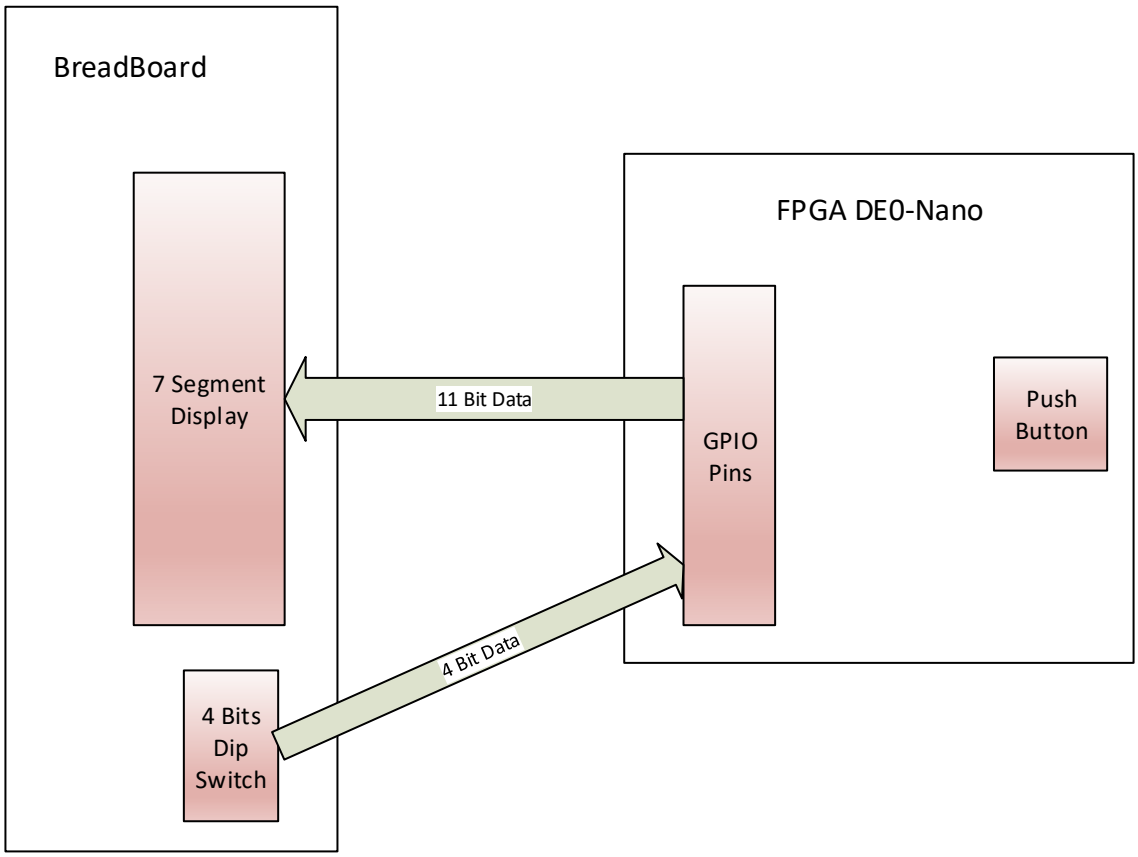
Constructing a 0-1 switch by using DIP-switches
==========
A dip-switch is an on-off switch. We want to convert it to 0-1 switch, ie in one position it will send a zero and in an another position it will send +3.3 volts.
For a single switch it works as shown in the circuit below.



For the 4-switch dipswitch the circuit is given below.



The details can be found at http://www.electronics-tutorials.ws/io/input-interfacing-circuits.html


BreadBoard
7 Segment Display
4 Bits Dip Switch
FPGA DE0-Nano
GPIO Pins
Push Button
11 Bit Data
4 Bit Data