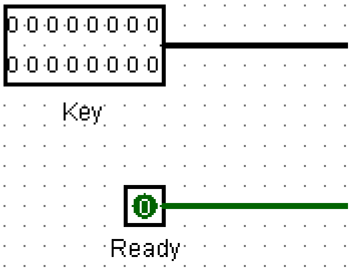
CSE4117 Microprocessors Homework II – Final Version

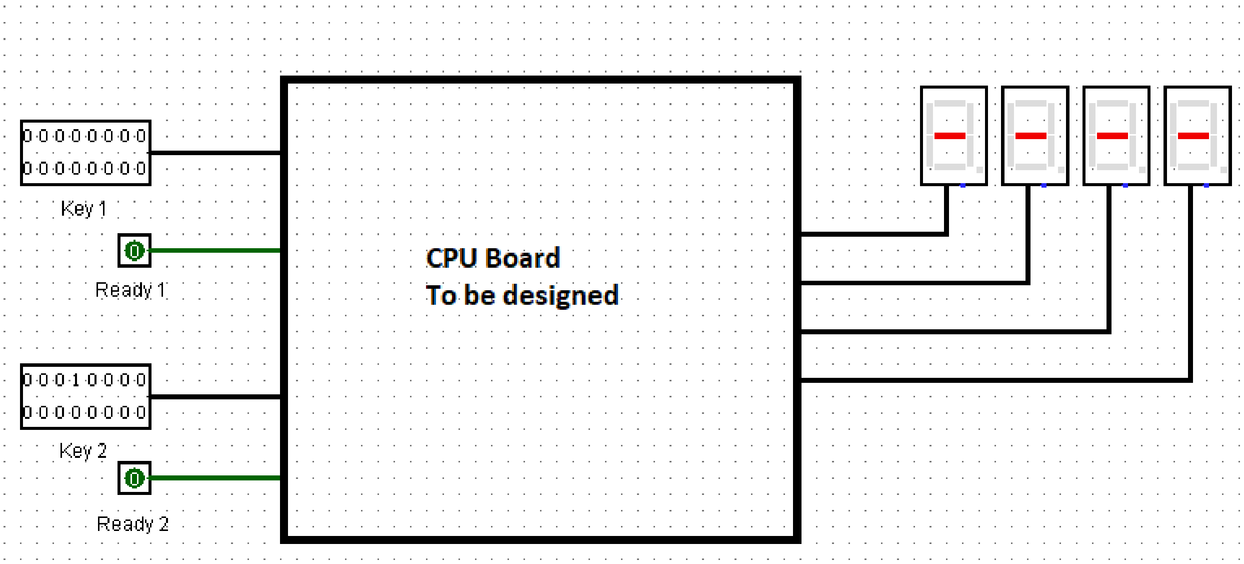
Due date: December 29, 2019 – Demo date: January 2-3, 2020

Q1) In this question, you will only use Logisim.

- You will use Bird CPU, 2 switch pads and 4 Hex Digit Displays and a memory chip.
- You will use polling as the I/O mechanism.
- For switch pad, use 16-bit Pin device for the switches and a single Pin device for the ready button. Pin devices should be chosen as 2-state from the Logisim menu.



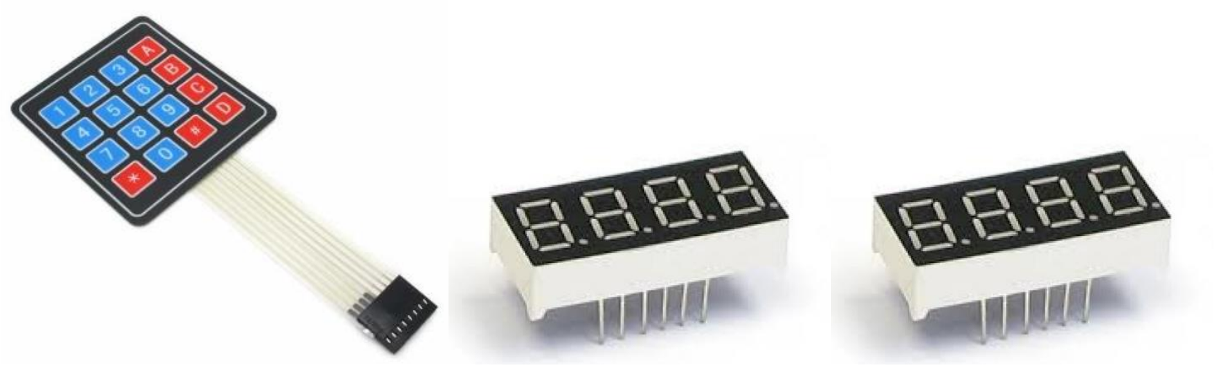
- Use Hex Digit Displays instead of 7-Segment display in Logisim (will make your life easy)
- The algorithm you will write for this system will work as follows;
  1. Start the Hex Digit Display at 0.
  2. Poll switch pad 1. If there is new data, read it and add it to the number currently written on the Hex Digit Display and display the result.
  3. Poll switch pad 2. If there is new data, read it and add it to the number currently written on the Hex Digit Display and display the result.
  4. Goto step 2.



- In short, you should design the internals of the black box above.
- You should decide the memory size and the addresses of switch pads and hex digit displays.

Q2) In this question you will implement a “pocket calculator” which can perform integer arithmetic on FPGA via Verilog by using bird CPU.

- All I/O must be performed by polling.
- You will take your input from a 4x4 Membrane Keypad (Shown in the figure below)



- In the keypad;
    - The keys 0-9 work in the usual way,
    - Key ‘A’ acts as ‘+’, i.e., addition
    - Key ‘B’ acts as ‘-’, i.e., subtraction
    - Key ‘C’ acts as ‘*’, i.e., multiplication
    - Key ‘D’ acts as ‘/’, i.e., integer division.
    - Key ‘*’ acts as ‘=’, i.e., display the result.
    - Key ‘#’ acts as clear button.

  These operations must work in the same way with a usual pocket calculator. Assume no overflow occurs.

- The result must be displayed on two 4-digit seven-segment display. (Use Common Cathode)
- Inputs and outputs must be decimal.
    - In input do the conversion by following this example;
        - If we want to enter the number is $k_1k_2k_3$ i.e., the digits k_1, k_2, and k_3 pressed in succession, and k_1 is the most significant and k3 is the least significant digit
        1. start with Value=0, i=1
        2. read k_i from the keypad
        3. Value= Value * 10 + k_i
        4. i=i+1, goto 2

You should do the output in reverse i.e., derive the digits k_3, k_2, and k_1 from a given number and display them in 7-Segment Display.

- In order to save ourselves from overflow conditions, we assume that the numbers we have entered and the results can always be represented by 16-bit numbers.
- No negative number can be entered from the keyboard. But the result of the operations can be negative. In that case, display the result with a minus in front of it.

Extra Credit: If you manage to extend the data bus of the CPU to 32-bits you will get +30 extra credits.