# 4CS015 – Workshop #5 TO BE SUBMITTED AS PART OF YOUR PORTFOLIO – please note this workshop covers material from lectures 3,4,5 and 6 (next week).

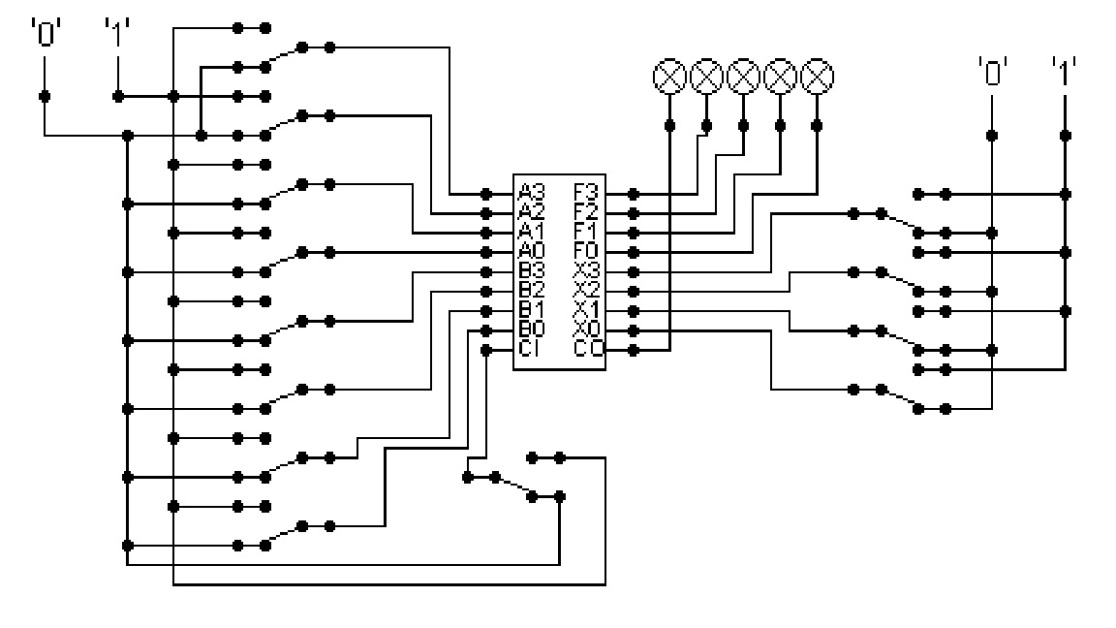
Name:

Student ID:

This is a marked workshop. It forms the second part of your portfolio. You will need to complete the workshop and then submit a copy of this document with a title that follows the following format (“DENNETT 1234567 wsp5.docx”), via CANVAS, by the deadline.

**Workshop tasks:**

Arithmetic Logic Unit

Load the LogSim Arithmetic Logic Unit Circuit **alu.cct** from inside the logsim application (You'll find it in the logsim folder) (***You may need to right-click on the link to download the file instead of opening it in the browser)***. It should look like this:  
  
  
  
The circuit behaves like a simple arithmetic logic unit. The inputs A0-A3 represent a 4 bit binary number. Inputs B0-B3 represent another binary number. A0 and B0 are the least significant bits respectively. The following table details the functions supported by the chip. All other control lines = 0.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Function | AND | OR | XOR | NAND | NOR | NOT A | ADD | SUBTRACT |
| X3 – X0 | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 1010 | 1011 |

Use A= 11 B=4, complete the following table in binary ***(10 marks)***:

|  |  |
| --- | --- |
| FUNCTION | OUTPUT |
| AND | 0000 |
| OR | 1111 |
| XOR | 1111 |
| NAND | 1111 |
| NOR | 0000 |
| NOT A | 0100 |
| ADD | 1111 |
| SUBTRACT | 0111 |

The logical operations are bitwise. Manually prove each operation has returned the correct result by  ***(15 marks)***:  
Example:  
 1 0 1 1

0 1 0 0 AND OPERATION

0 0 0 0 RESULT

1 0 1 1

0 1 0 0 OR OPERATION

1 1 1 1 RESULT

1 0 1 1

0 1 0 0 XOR OPERATION

1 1 1 1 RESULT

1 0 1 1

0 1 0 0 NAND OPERATION

1 1 1 1 RESULT

1 0 1 1

0 1 0 0 NOR OPERATION

0 0 0 0 RESULT

1 0 1 1 NOT A OPERATION

0 1 0 0 RESULT

1 0 1 1

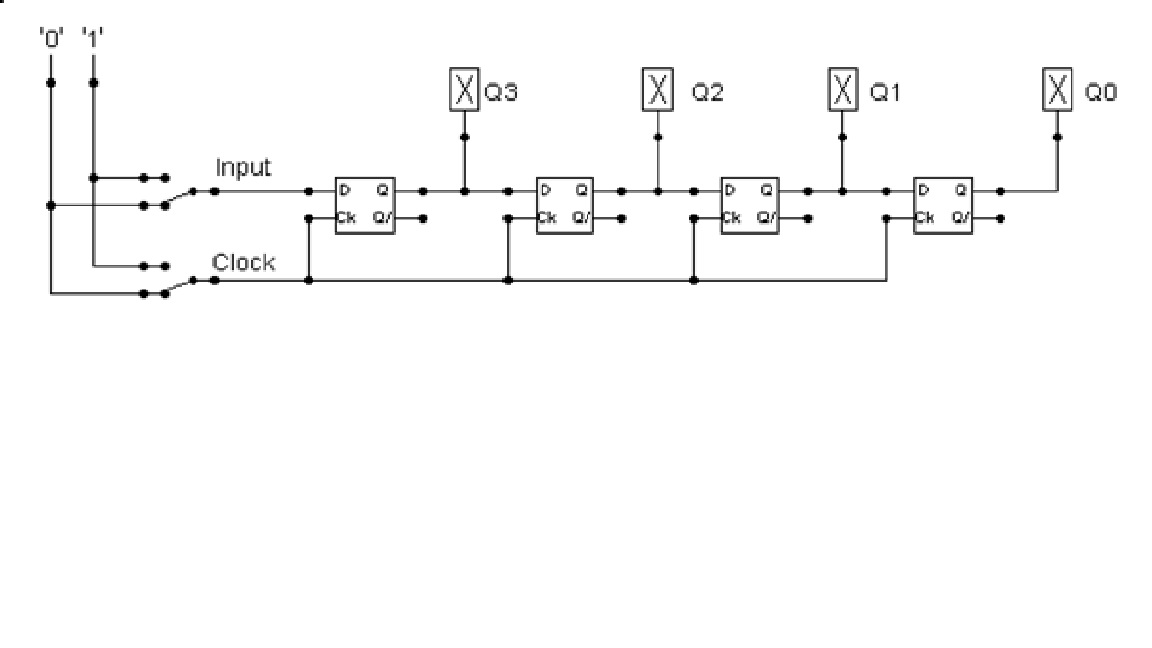
0 1 0 0 ADD OPERATION

1 1 1 1 RESULT

1 0 1 1

0 1 0 0 SUBTRACT OPERATION

0 1 1 1 RESULT

Lecture **6** material: Serial to Parallel Decoder:  


Build the circuit above and complete the following timing diagram by filling in the table spaces with ‘1’ or ‘0’. ***(15 marks)***

***Remember to disable preset / clear for the d-type in properties.***

***Work from left to right for the inputs, when you run your logic circuit get the output at 1111 with 10 input, this will be you first input column)(remember you first gate is Q3 which is at the top of the table). You may need to clock 0-1 a few time when the circuit is running to get the output required. Note where Q3-Q0 are on the table and on the logic circuit.***

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Input | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
|  | oldest |  |  |  |  |  |  |  | newest |
| Clock | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| Q3 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| Q2 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |
| Q1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| Q0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Describe what the circuit does. ***(15 marks)***

****Initialization**: Initially, all D flip-flop outputs (Q3-Q0) are high (1). This is represented by the "oldest" state in the timing diagram.**

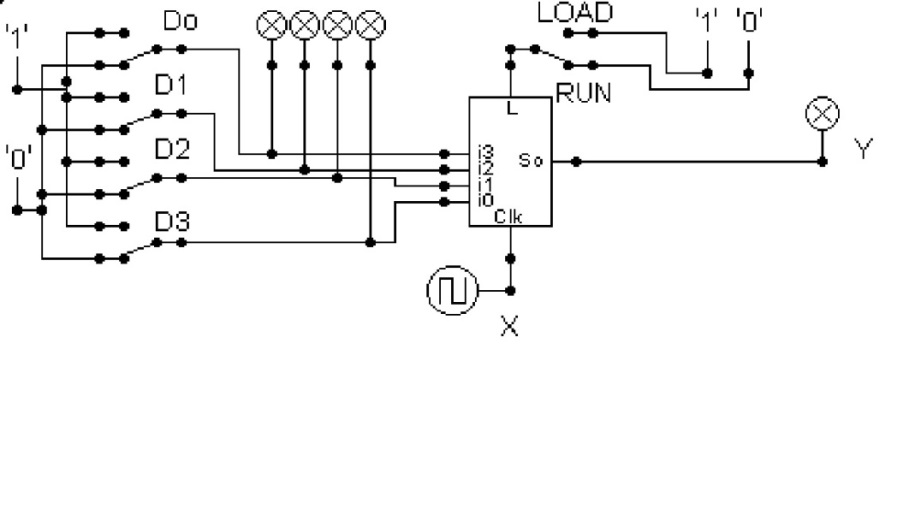
****Loading Data**: To load data into the shift register in parallel, you need to set the PL input high and provide the desired data at the D inputs of each flip-flop. When the clock signal transitions from 0 to 1 (rising edge), the data at the D inputs will be transferred to the Q outputs of the flip-flops. This operation is represented by the "newest" state in the timing diagram.**

****Shifting Data**: Once the data is loaded into the flip-flops, subsequent clock pulses (0 to 1 transition) will cause the data to shift to the right. The output of each flip-flop (Q) will take the value of the input (D) of the previous flip-flop. This is shown in the timing diagram as the clock transitions from 0 to 1, shifting the data from oldest to newest.**

Combining everything we have covered so far (lectures 3-6).

Parallel to Serial converter

Open the LogSim circuit **week5.cct** from the Logsim folder. Also see week5 v2.cct for more help. It should look similar to this:

  
Describe what this circuit does. ***(15 marks)***

Design and add to the above circuit an additional circuit that takes the Clock X and the Output Y and decodes Y into 4 output indicators so that they match D0 – D3. Insert the LogSim GIF output of your design in the space below.

The highest marks will go to those who design the circuit such that it **AUTOMATICALLY** stops (not pauses) when the input to the circuit matches the output to the circuit

*Note: Save your GIF image when your output indicators match the input D0 - D3*. ***(30 marks)***

