ISTANBUL TECHNICAL UNIVERSITY

COMPUTER ENGINEERING DEPARTMENT

BLG 222E
COMPUTER ORGANIZATION

PROJECT 1 REPORT

GROUP NO : 1

GROUP MEMBERS:

150170095 : Mehmet ALTUNER
150170026 : Berkay OLGUN
1500170103 : Atahan ÖZER
150170076 : Ekrem UĞUR

SPRING 2020

1 Part-1

First part of this project is formed of two subsections. In this part we were expected to design registers using logisim software. We designed an 8 bit and a 16 bit register structure using logical gates, multiplexers and D Flip-Flops.

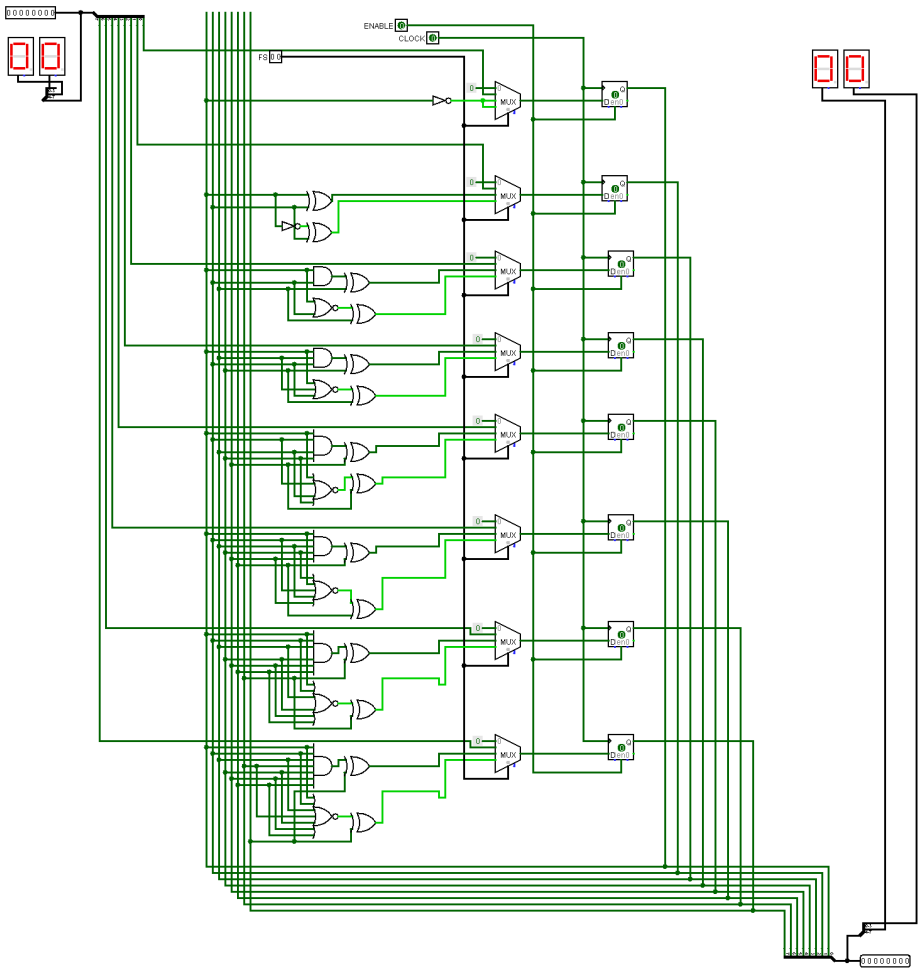

Figure 1: 8 bit Register Structure

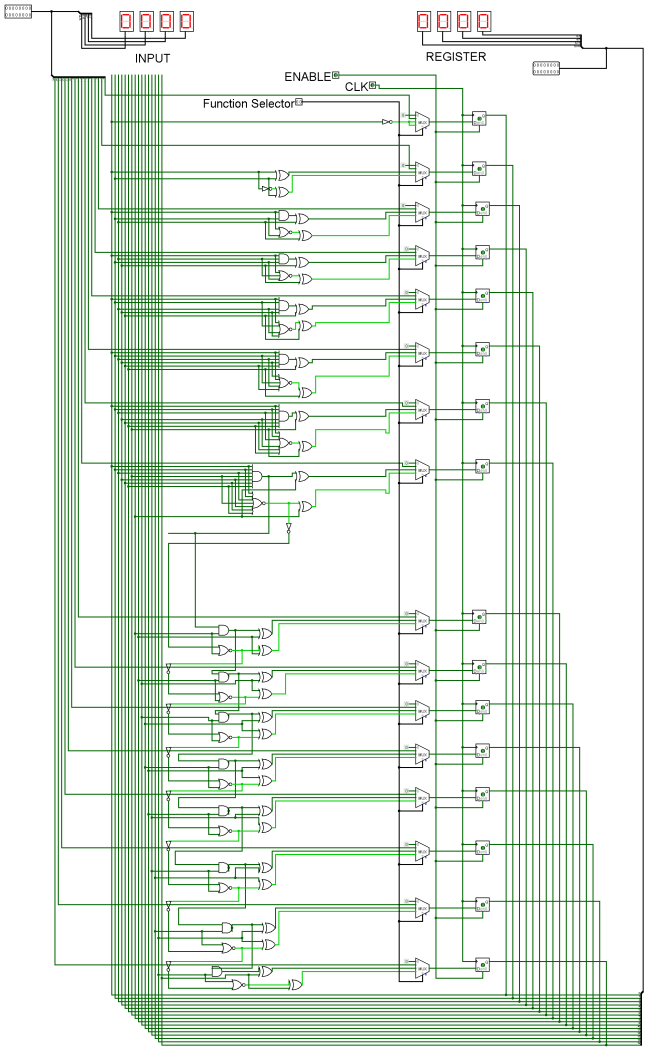

Figure 2: 16 bit Register Structure

Both registers share the same functionality, which is provided via two bit input signal function selector (FunSel). If the register is enabled, FunSel decides which action register will take in the next rising edge of the clock cycle.

E	FunSel	Q^+
0	ϕ	Q
1	00	0
1	01	1
1	10	Q + 1
1	11	Q - 1

Figure 3: Function selector signal / Output table

To achieve this kind of functionality, we used multiplexers before each D Flip-Flop. 00 meant a constant low signal, 01 meant the corresponding input bit would be selected in the multiplexer. While executing increment command, n^{th} bit of the output was calculated via combining first n-1 bits in a LOGICAL AND gate, and giving the output of this gate to a LOGICAL XOR gate together with the initial value of n^{th} bit. A similar scenario was staged to execute the decrement command. With only a difference of using LOGICAL NOR gates instead of LOGICAL AND gates.
Finally, in order to use these registers in the upcoming parts and projects, we designed the subcircuit representations of the registers we implemented and completed this part of the project.

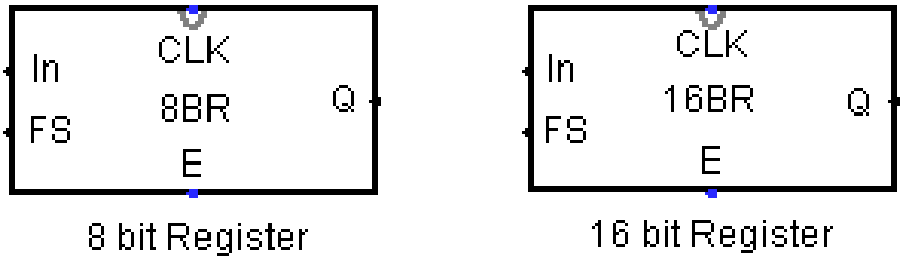

Figure 4: Subcircuit models of both registers

2 Part-2

In the second part of the project, we implemented two register banks with 4 registers and 3 registers respectively. We used the registers we have designed during the first part of the project.

2.1 Part-2a

In this part, we designed a register bank with 4 registers. We simply used the register we have designed in Part-1 and connected to them FunSel, RegSel, Input, OutCSel and OutDSel inputs. FunSel selects which function to use and RegSel selects which register to operate on, note that multiple registers can be selected at the same time. Input is an 8 bit data. OutCSel and OutDSel selects which registers' output is shown on, respectively, OutC and OutD. We implemented this using two multiplexers. OutC and OutD is shown using two hex digit displays.

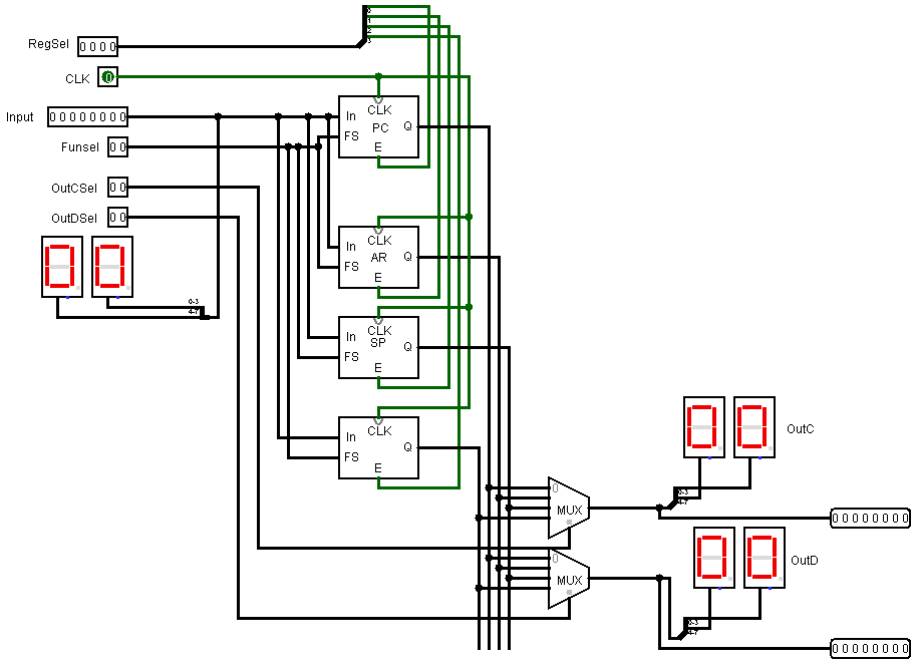

Figure 5: 4 Register Bank Circuit of Part-2a

2.2 Part-2b

In this part we designed a circuit similar to the first part of the part 2. We connected 3 register banks to 2 multiplexer's to show outputs on Hex-Digit

Display. Similarly we have 4 bits RegSel for enabling input on registers, FunSel that selects the function to do on the registers, an 8 bit input to apply and 2 selection lines OutputC and OutputD, to choose the register we want the show the output.

According to input given in OutputC or OutputD selection lines the multiplexers select the correct register (if enabled) to show on the outputs.

s_0	s_1	Output
0	1	PC
1	0	AR
1	1	SP
a	a	PC

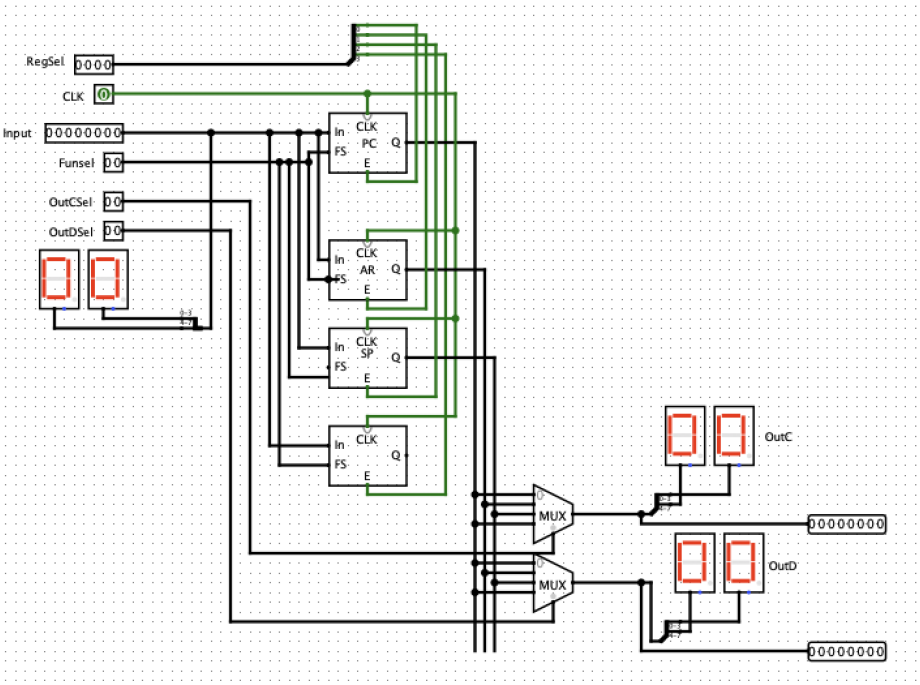

Figure 6: 3 Register Bank Circuit of Part-2b

2.3 Part-2c

In this part of the project we have designed an 16 bit IR register.In order do that we used the 16 bit register which we build in part -1b.3 problem occurred

while using the old 16 bit register.The first problem was that old register had 16 bit input but our new register should have 8 bit input.In order to fix that we connected same 8 bit input into upper 8 flip-flop and lower 8 flip-flop so that we got 16 bit register with an 8 bit input.Second problem were the loading of the flip-flops since they are connected to same input we had to load them one by one which means when upper flip flops are enabled lower flip flops must be disabled or vice versa.We provided that by using a switch circuit(figure 3) which is only enables either upper flip flops or lower flip flops when function selector is equal to 11.

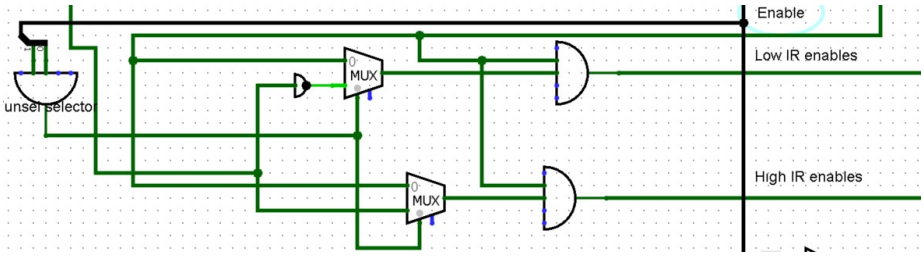

Figure 7: D flip flop enable switcher

The circuit in figure(3) takes 3 "inputs function selector ,enable and L/H" .The 2 bits of function selector are fed to an AND gate and output of the AND gate are connected to switch port of the multiplexers so that when there is no load signal is coming normal enable signal will be connected to enable ports of flip flops.When load signal is coming from function selector, multiplexers will switch to L/H signal instead of normal enable signal.The outputs of the multiplexers are connected to an AND gate with normal enable signal so that they only work when enable is 1. L/H signal is inverted one of the multiplexers so that when signal is low, flip-flops of the lower bits will get high as enable and higher bits get low and vice versa.