**Data Path**

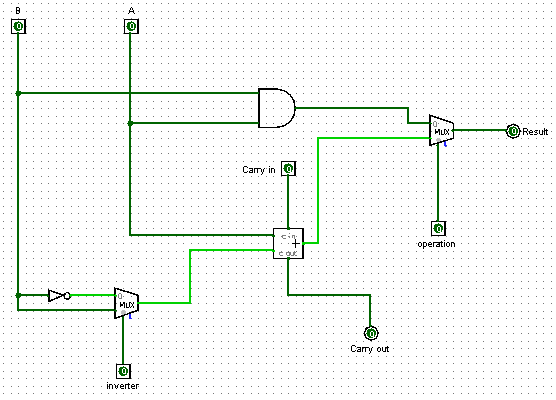
**1.ALU**

Our CPU is designed in a way that it compute add ,sub,shift and and operation

In our design we divided this operation to two parts

**Part one : add , sub, and operation**

For this we build 1 bit circuit and use it to the 8 bit as follow



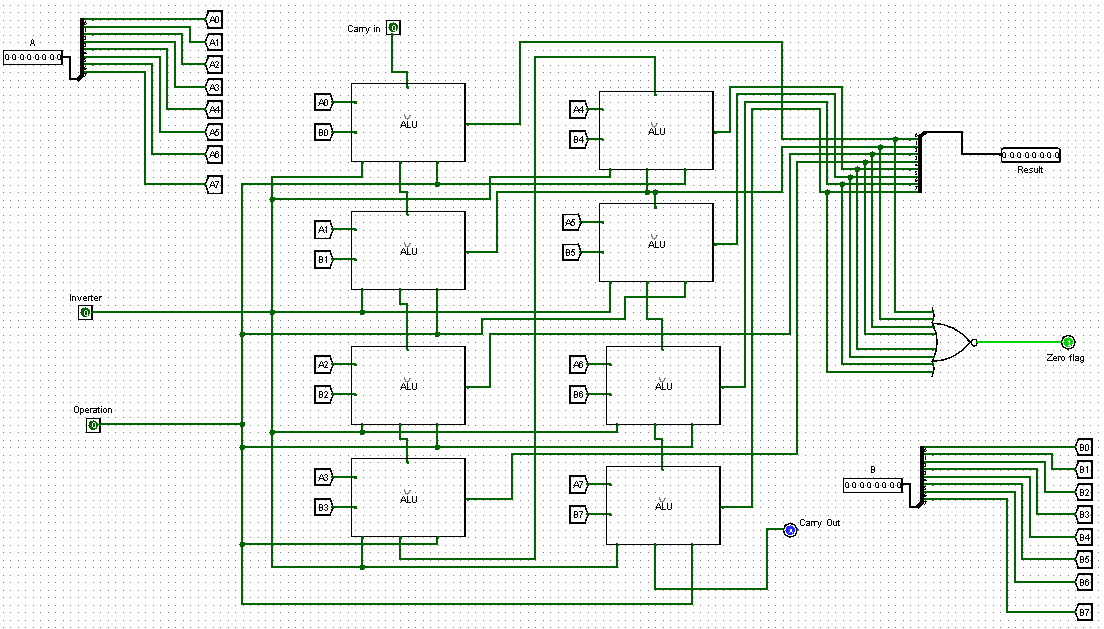
When inverter bit is 0 it complement the value of b and when the carry in bit is one it we obtains the 2’s complement of b then adding it to is equivalent to subtracting B from A

A+(-B)=A-B

And when the operation bit is 1 we have either A+B or A-B depending on the value of inverter and the carry in bit .and when the operation bit is 0 we select the AND operation so it compute

A and B .

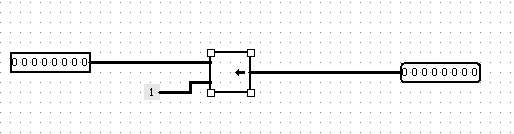
And we build the sub ckt to 8 bit using the single bit of above as follow



-zero flag is high when all the value of computation is zero this is why we use nor gate with 8 input which is used in conditional branch instruction.

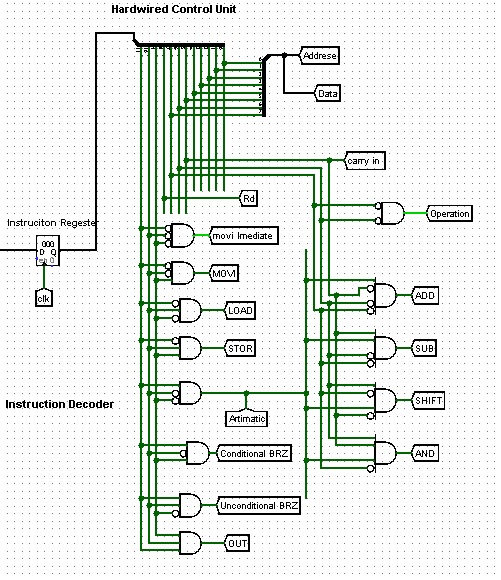
**Part two: shift operation**

For this part we use logical shift to left by one bit .for input Rd either its register A or B depending on the instruction



**2.CONTROL UNIT**

For our control unit we implement it using hardwired approach and decode each instruction fetched from instruction register using modified AND gate as follow



We used splitter to split the hexadecimal to 12 bit binary sequences then the 3 MSB is examined for each opcode for instance instruction load have opcode of 010



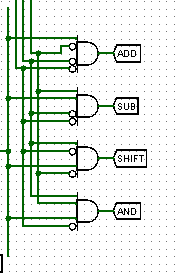
so we invert the first and the third bit and give it to AND gate then if the value is high the Load flag will be high and Load data path is executed and the same procedure is followed for each instruction set .

For Arithmetic instruction we have to check two things first if it’s arithmetic or not then which arithmetic operation is chosen based on the functional code after the register specifier bit Rd

For arithmetic operation the opcode is 100 so if we inverte the second and third bit and input it to AND gate we can check either its arithmetic instruction or not



If the arithmetic flag is on then we check which arithmetic is chosen using this logic with the fourth bit will be the flag from arithmetic opcode and the three bits are the functional codes



-For the out ,conditional and unconditional branch we follow the same procedure .

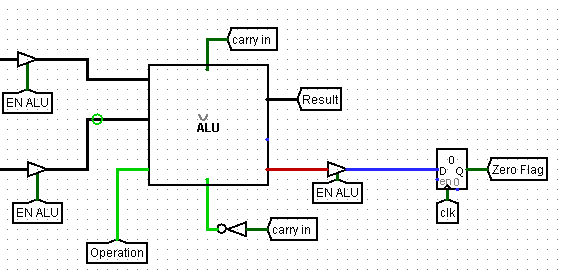
And the operation flag is taken from the 5th and 6th (count form left to right) from instruction note that in ADD and SUB instruction this two bit are both zero so we can use two input modified AND gate when this flag is high and or subtract is chosen depending on the carry in bit that we defined as the last 7th bit(count form left to right) in the instruction from instruction the seven is 0 for addition and 1 for subtraction using this logic we can implement either the add or subtract operation.

**Instruction sets**

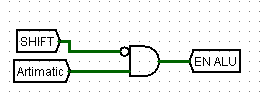
**1.ADD ,SUB,AND instruction**

For add instruction the arithmetic flag, ADD flag and operation flag is high and the carry in bit is low. Rd flag is set o for A or 1 for B

Feeding this value to the ALU we implement the add operation .



EN ALU :(enable the ALU flag) used since we divide the arithmetic operation to two either shift or add,sub,and so this flag check if its not shift operation (since shift operation is implemented using other block )and it’s arithmetic then it enables the ALU.



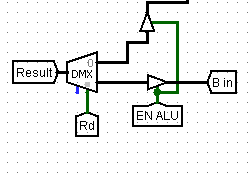
And the value of zero flag is changed if the operation is arithmetic and not the shift logic or other and we used one bit register here to hold the value in case we have conditional branch instruction in our instruction memory.

* We feed the inverted carry in to inverter pin since 1 signifies no complement register B value so we perform addition instruction .from the first figure.
* If the carry in bit is 1 then the inverted carry input signify the 2’s complement of B and the operation bit chooses sub instruction so subtraction is performed .
* For AND instruction the operation flag will be low since the 5th and 6th bit are 0 and 1 (counted from left to right) so AND operation is performed.

And finally depending on the value of Rd the right register is used to write the Result of the operation

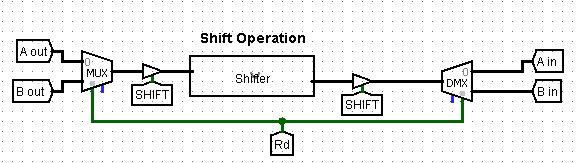
Rd=0 the value is written in register A

Rd=1 the value is written in register B



**2. SHIFT instruction**

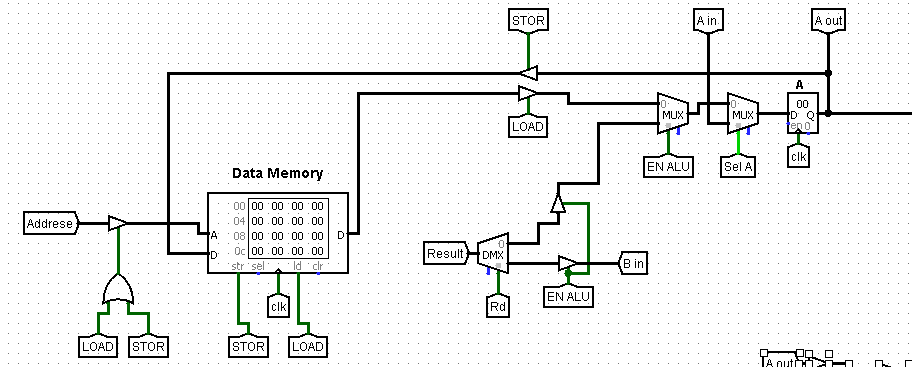
For this instruction the shift flag is on so shift operation is performed



Depending on the value of Rd we chose which register value to shift and store se we use multiplexer and de multiplexer with selector bit of Rd.

**3.LOAD and STOR instruction**

* For load instruction we load the value in given address to the register A as specified in the design question.so the load flag is high so using the address we load the value to register A .

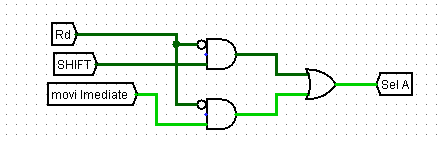


The 8 bit address value is feed to data memory and either we load or store value to register or data memory this is why we use controlled buffered with LOAD ored with STOR flag.

since the load flag is high ,data value from specified address is loaded to register A

since the value of EN ALU and sel A is zero at this point .

**sel A.**



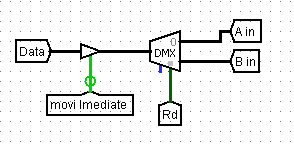
This bit specify if we write the value from shift or move immediate instruction if it’s high

Ain is selected as input for Register A.

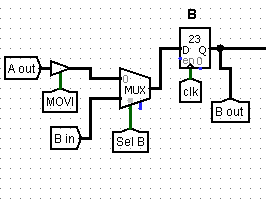
* For Store instruction we store the value of register A in the given address since the store flag is high and address buffer set to one .

**4. MOVI immediate and MOVR instructions**

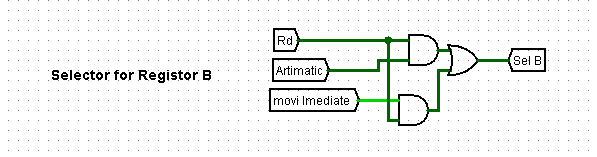
for move immediate instruction the move immediate flag is set high then the 8 bit immediate value (data) is loaded to register A or B depending on the Rd value.



* For movr instruction MOVI flag is high and the value in register A is copied in to register B.



**Sel B**



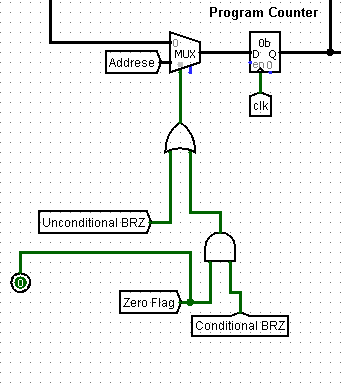
When the instruction is not either arithmetic or move immediate and the value of Rd is not 1 the selector bit select the input at o since sel B is zero then the data is loaded to register B from A.

**5. BRZ and BR instructions**

This two instruction modify the value of program counter

For BRZ (conditional branch to specified address location) when the zero flag is set high .

For this instruction the conditional BRZ flag is set high and the program counter loaded with the value 8 bit address from 12 bit instruction .

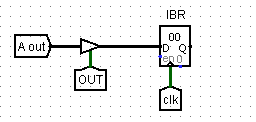


For conditional branch we check if the zero flag is high then load the program counter with the value of address this is why we use AND gate.

For unconditional branch instruction we only check if the unconditional branch flag is set if its high then load the address filed into the program counter .

**6.OUT instruction**

When the out flag is set high the value in register A is written to IBR



**---------Thank you -------------**