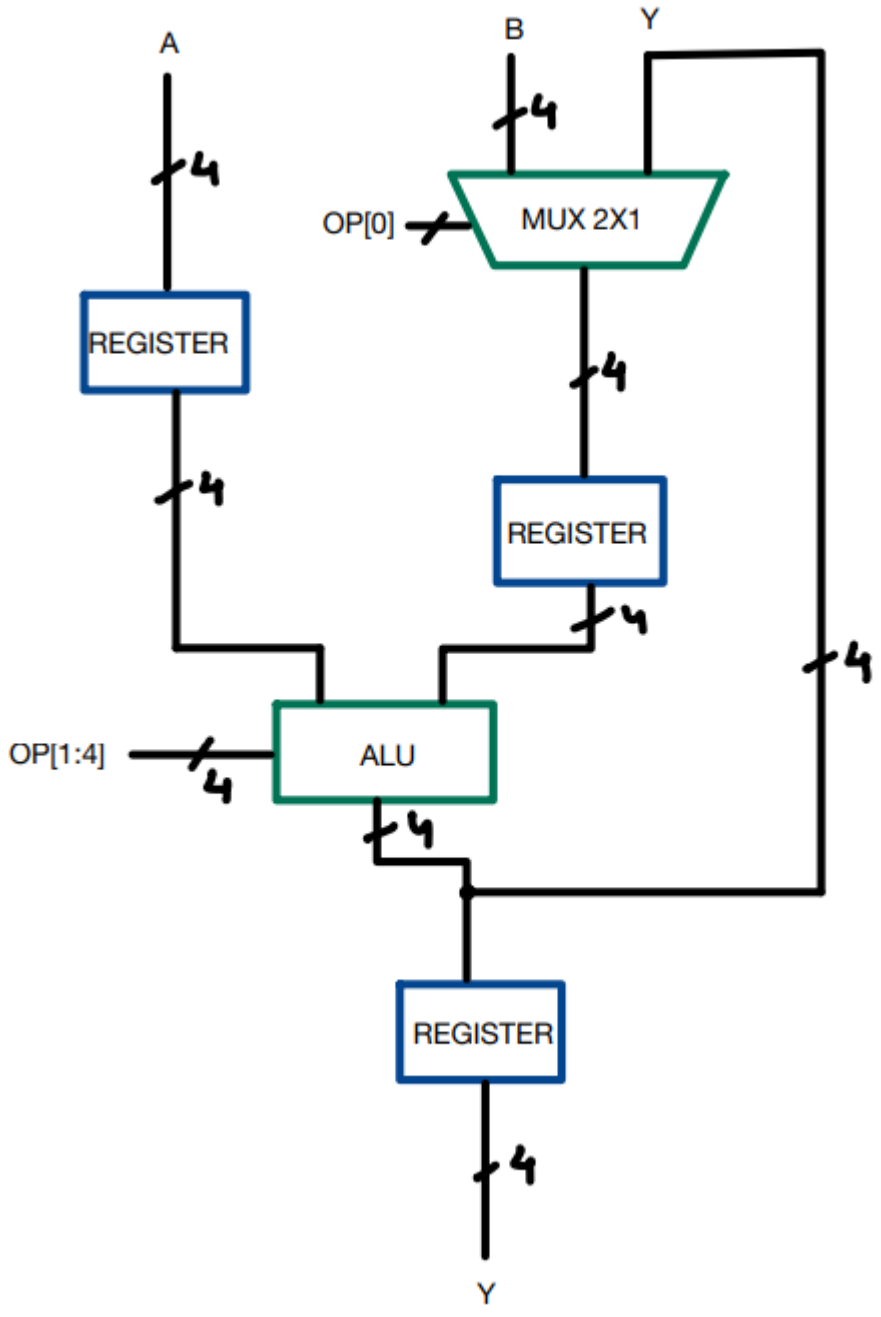
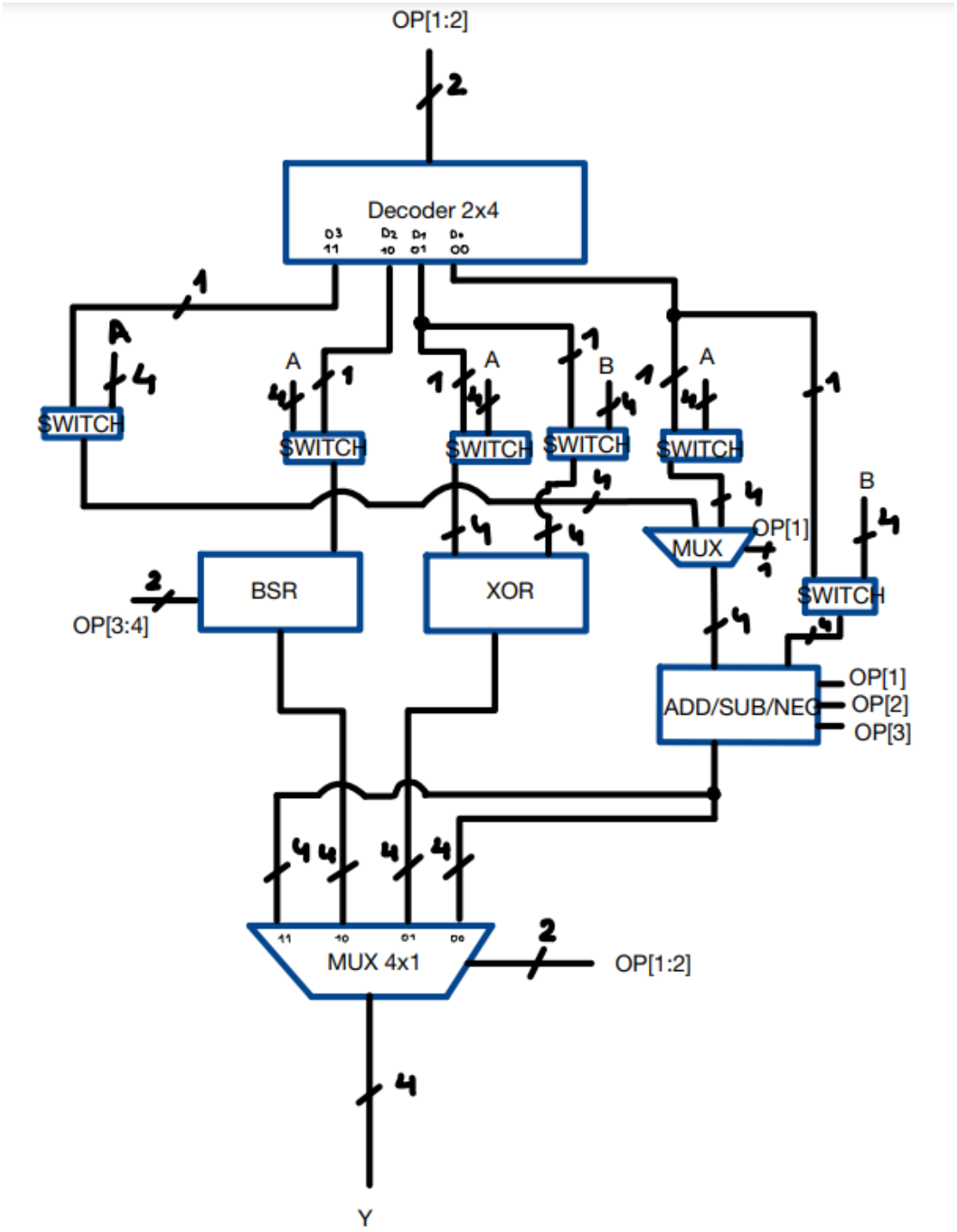
VLSI Final Project
4-bit ALU Design

Phase 1
Floor Plan

The Block Diagram:

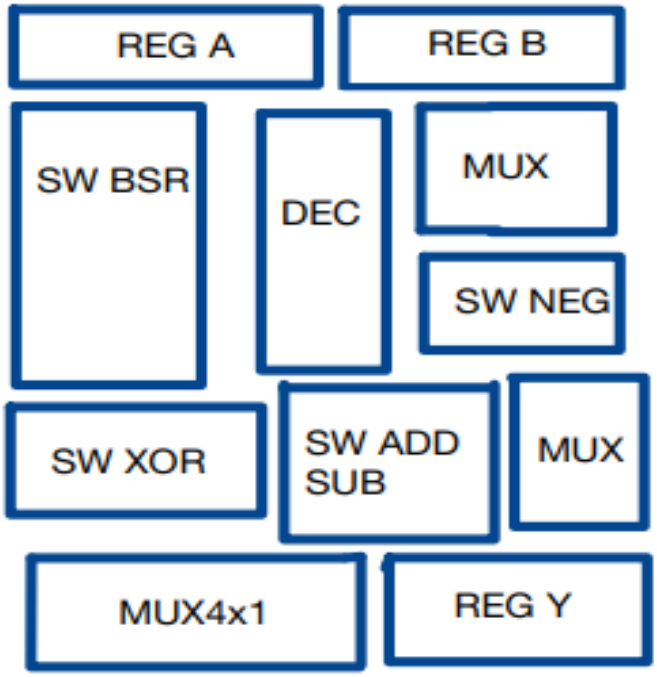


In the following ALU diagram we show more of the implementations' specifications:



These switches are the AND2:1 gates we will specify in the following pages; and it represents 4 of them, which means every switch has 4 AND 2:1 gates, and the gates work by doing bitwise between the input bits and the corresponding bit from the decoder output (like doing AND between the inputs A0, D0 and also for A1, D0…).

Floor Plan(הרצפה):



Brief Description of the Block Diagram

We have two 4-bit Inputs provided from Registers: A and B. The input B is multiplexed between 4-bit external input (Register B) and the output of the ALU (Y).

Four D-flip flops are used as Registers to store A and B. The output of the circuit is 4-bit number Y and is fed back to the multiplexer as a choice input as shown in the Block Diagram.

D-Flip Flops act as a Register to store o/p. The D-Flip Flops are positive level triggered.

Working process of the ALU

The 4-bit input A and B are fed into the ALU to perform the various algorithmic operations. Inside the ALU, the operations which must be performed are controlled by a 5-bit OPCODE. The specification of the OPCODE is as follows:

* The first row is to indicate the number of the bit in the opcode while the second row is to indicate the "position" of the bit in the opcode.

5^{th} BIT	4^{th} BIT	3^{rd} BIT	2^{nd} BIT	1^{st} BIT
4	3	2	1	0

The first bit (in position 0) of the OPCODE is fed to the mux(2x1) in the entry) as a selection bit between B and Y.
The 2^{nd} and 3^{rd} bits are fed to a 2X4 decoder to select an operation.
The 4^{th}, 3^{rd} and 2^{nd} bits are used to be responsible to the type of operation we choose to perform(ADD/SUB/NEG) . While performing Barrel-Shifter-Right The bits (in position 4 and 3) are used as an indicator the amount of shift required by the user.

The Knowles [2,1,1,1] Adder is being used to perform ADD/SUB/NEG. The XOR gate inside ADD/SUB block complements the 4-bit number B and sets Cin = 1 to perform SUB (shown later in this file).

The OPCODE assignment

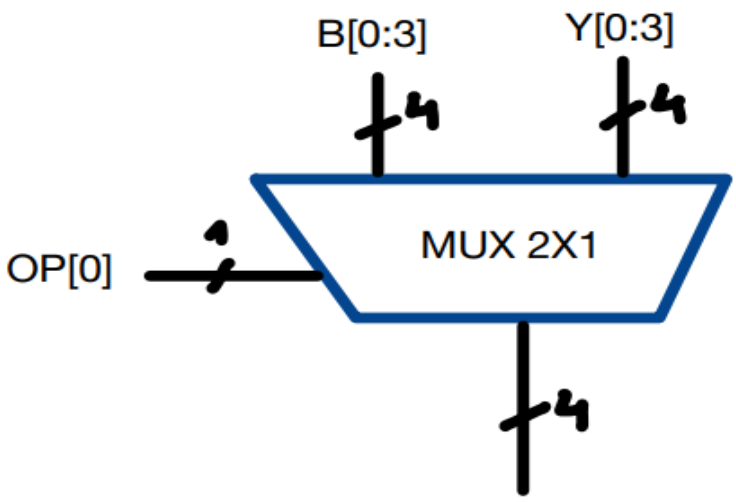
	operation	Bit4	Bit 3	Bit 2	Bit 1	Bit 0
ADD/ SUB/ NEG	ADD	X	0	0	0	
	SUB	X	1	0	0	
	NEG	X	1	1	1	
BRS	Shift 4	0	0	1	0	
	Shift 1	0	1	1	0	
	Shift 2	1	0	1	0	
	Shift 3	1	1	1	0	
Logic	XOR	X	X	0	1	

- If the OPCODE first bit (in position 0) is 0 we will take input B
- If the OPCODE first bit (in position 0) is 1 we will take input Y

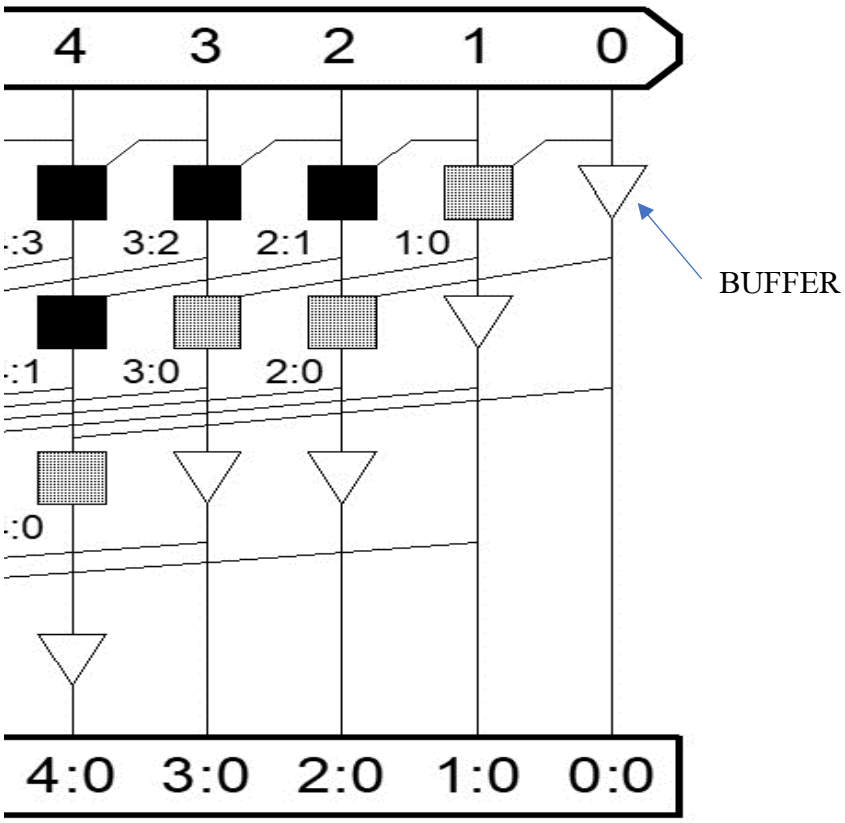
Basic Block Design (logic)

4-bit MUX 2:1

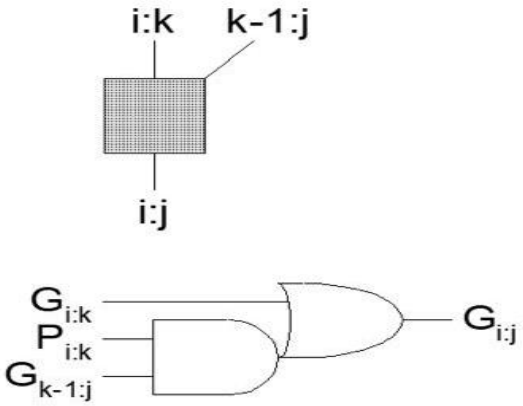
The first bit (in position 0) OPCODE bit determines the input (B or Y)



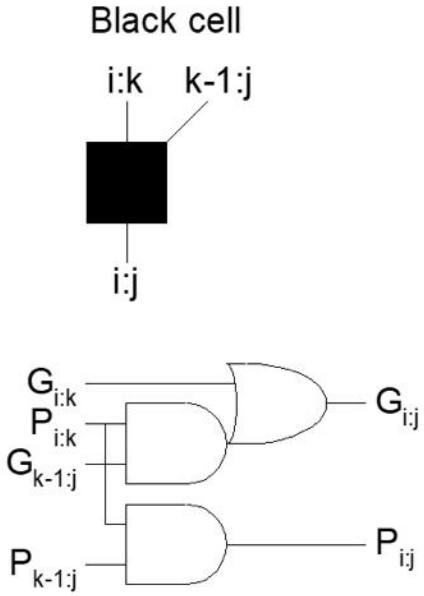
Knowles [2,1,1,1] Adder



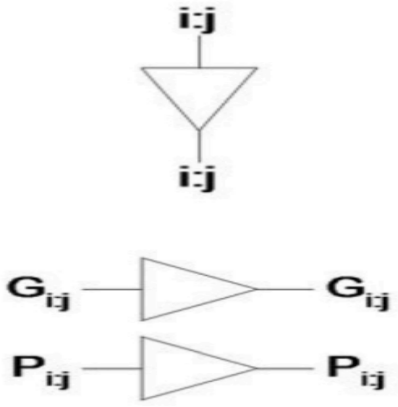
Grey cell



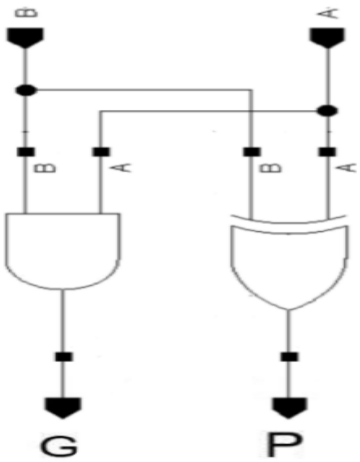
Black cell



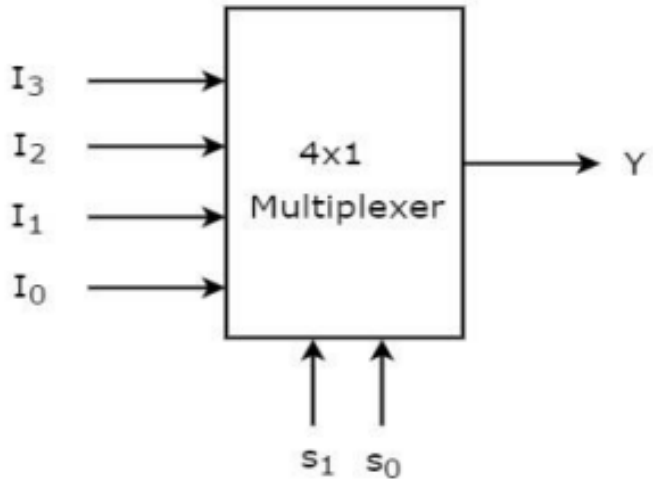
Buffer



GP-cell

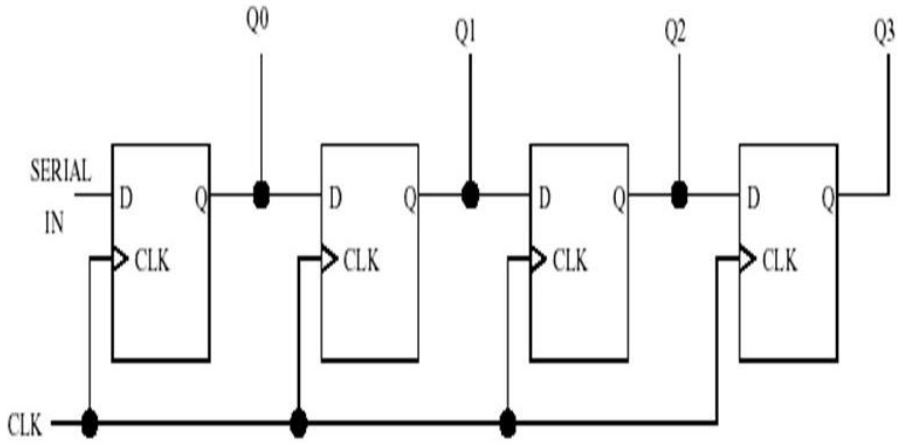


4-bit Mux 4:1

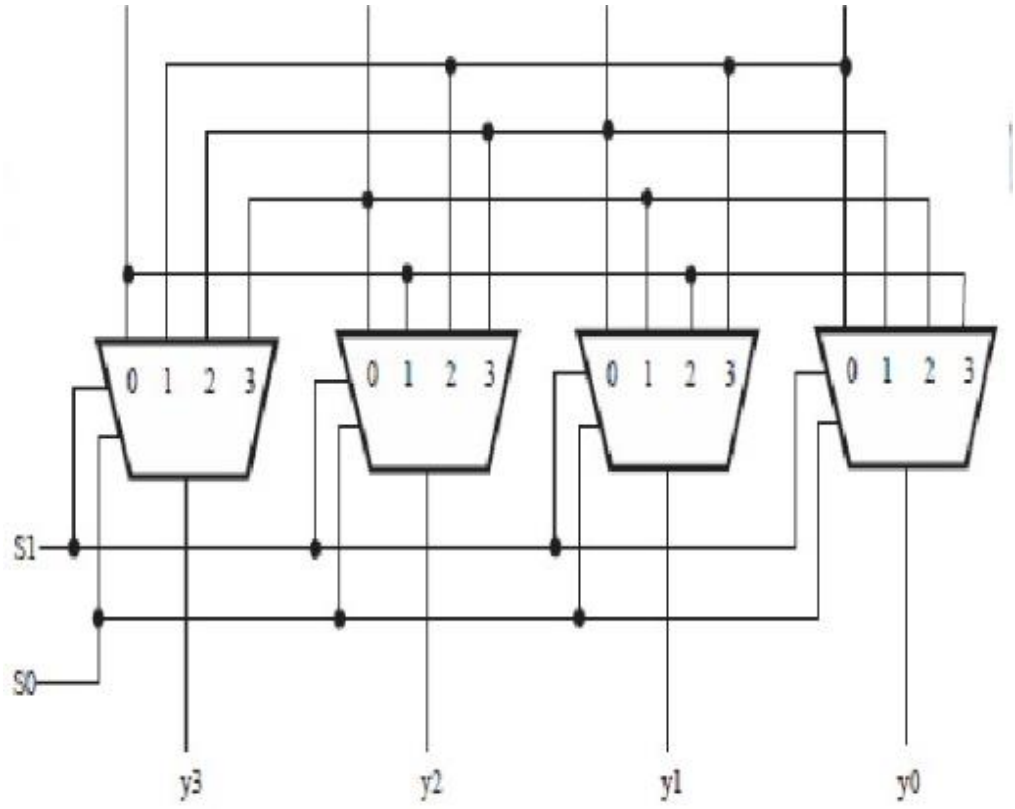


4-bit Register

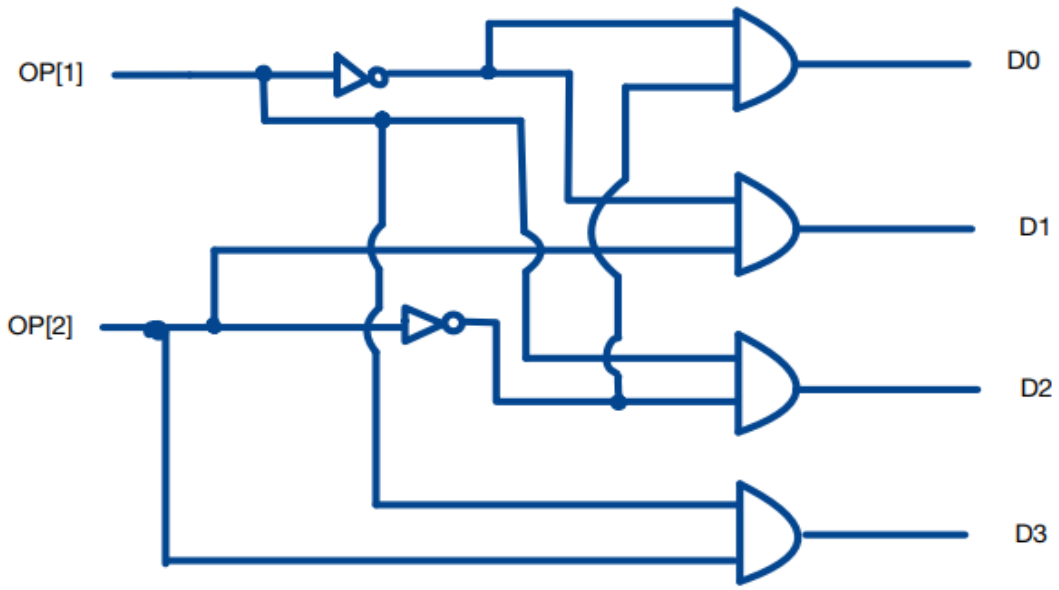
We have 3 components like this for A, B and Y.



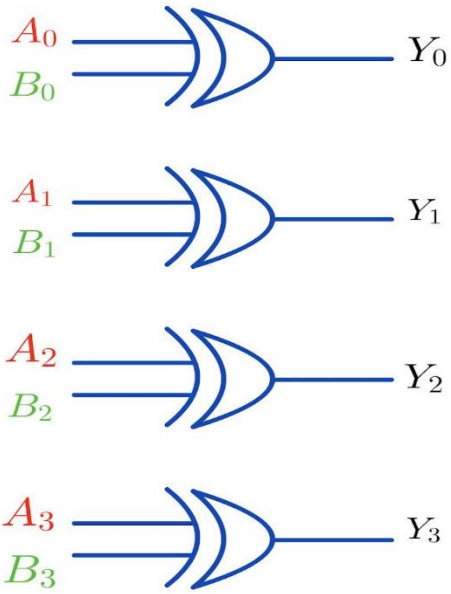
4-bit Barrel-Shifter-right



Decoder 4:2

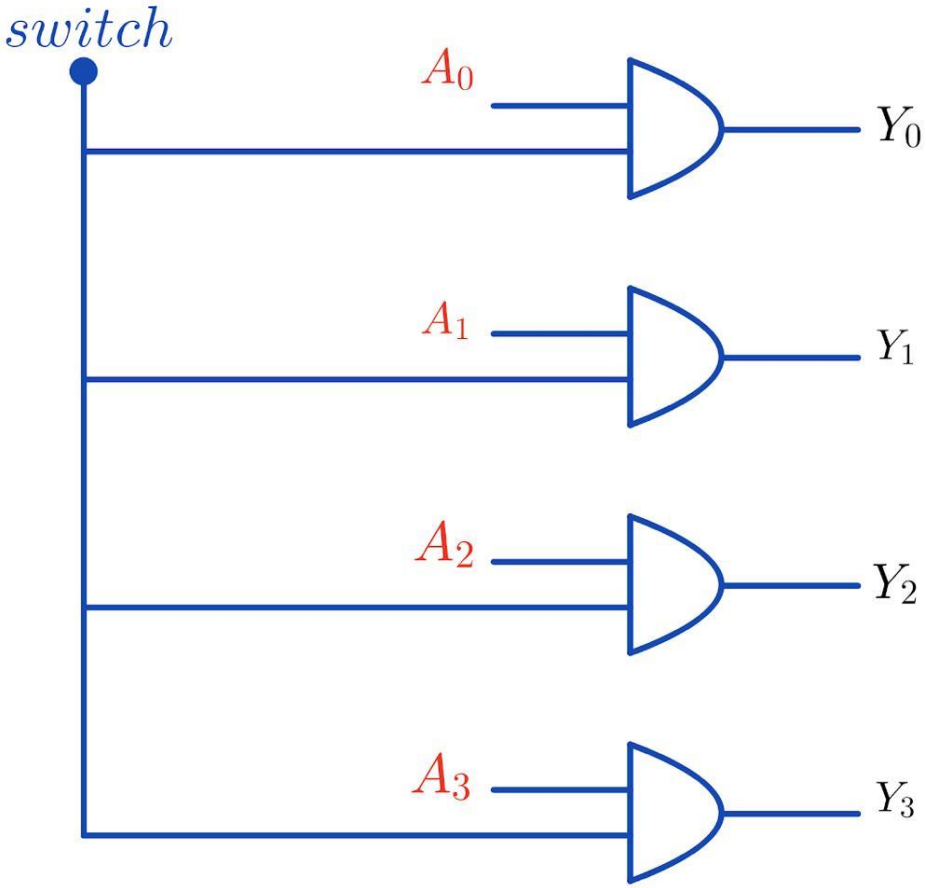


BITWISE XOR



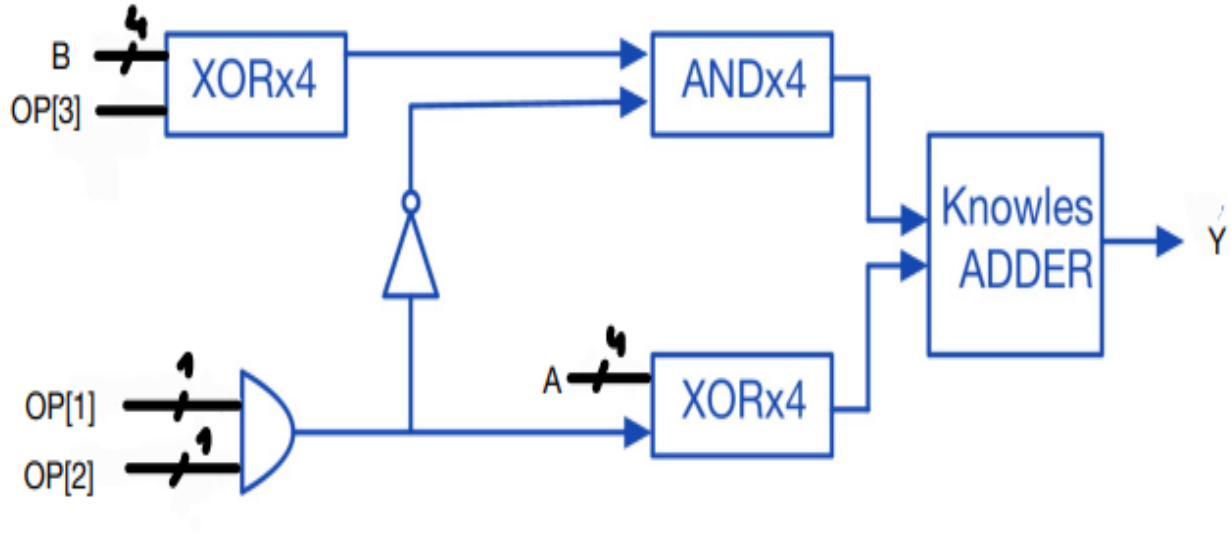
SWITCH

Switch For efficiency purposes we introduce switch component that enables targeted operations and disables undesired operations in each procedure. The switch block is made of 4 AND gates of size 2X1 , the inputs one of the inputs of each AND is the 1 bit long output of the decoder , and the other one is single bit from any of the inputs A,B or Y. For blocks that has single input we would implement one switch block , for instance NEG and BSR , whereas XOR and ADD/SUB would have two blocks of switches each containing 4 AND gates. The following diagram is the switch block:



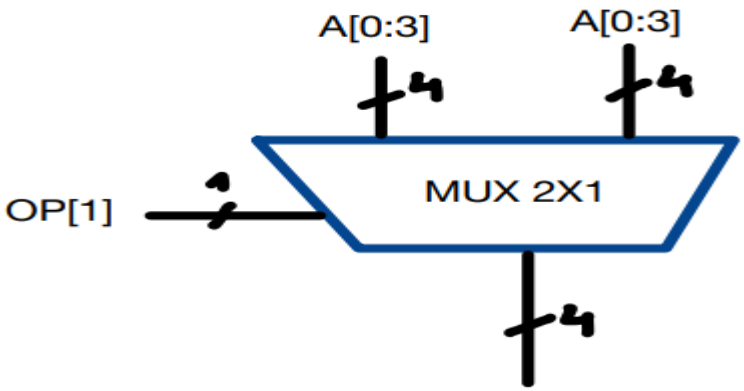
ADD/SUB/NEG

In the block that is responsible for adding subtracting and converting a number to be negative , for simplicity , we divided the functionality into two , one for NEG functionality and one for ADD/SUB , it can be seen that once we choose the opcode to direct us to use NEG we automatically disable B and we choose between A from the NEG path and A from the ADD/SUB path (which is zero in that case)



4-bit MUX 2:1

The second bit (in position 1) OPCODE bit determines which input A to choose , either for Negation or Adding / Subtracting , the input is determined by the first opcode (opcode[1])



Area Utilization

Component	Single Area $(um)^2$
AND 2X1	1.368
OR 2X1	1.368
XOR 2X1	2.736
INVX1	0.684
BUFFERX2	1.71
DFFHQX1	5.472
MUX 2:1	2.394
MUX 4:1	7.524

Component	Units	Total Area $[um]^2$
BITWISE XOR	4*XOR2X1	10.944
BSR	4*MUX4X1	30.096
SWITCH	4*AND2X1	5.472
DECODER 2:4	4*AND2X1+2*INVX1	6.82
BLACK cell	2*AND2X1+OR2X1	4.104
GREY cell	OR2X1+AND2X1	2.736
ADD/SUB	4* BLACK cell+4* GREY cell+5*BUFFERX2	35.91
REGISTER	4*DFFHQX1	21.888

Final calculation for area estimation

ADD/SUB + 3*BITWISE XOR + BSR + DECODER + 6*SWITCHES + MUX4:1 + 3*REGISTERS+2*MUX2:1 = 216.466 $[um]^2$

Area Utilization

Component	Single Area $(um)^2$
AND 2X1	1.368
OR 2X1	1.368
XOR 2X1	2.736
INVX1	0.684
BUFFERX2	1.71
DFFHQX1	5.472
MUX 2:1	2.394
MUX 4:1	7.524

Component	Units	Total Area $[um]^2$
BITWISE XOR	4*XOR2X1	10.944
BSR	4*MUX4X1	30.096
SWITCH	4*AND2X1	5.472
DECODER 2:4	4*AND2X1+2*INVX1	6.82
BLACK cell	2*AND2X1+OR2X1	4.104
GREY cell	OR2X1+AND2X1	2.736
ADD/SUB	4* BLACK cell+4* GREY cell+5*BUFFERX2	35.91
REGISTER	4*DFFHQX1	21.888

Final calculation for area estimation

ADD/SUB + 3*BITWISE XOR + BSR + DECODER + 6*SWITCHES + MUX4:1 + 3*REGISTERS+2*MUX2:1 = 216.466 $[um]^2$