PROJECT REPORT ON ALU

Name: Matam Basavaraju

M-ISS id: M-ISSDV0524

ABSTRACT

An arithmetic-logic unit is the part of a central processing unit that carries out arithmetic and logic operations on the operands(8-bit width) in computer instruction words(opcodes). The operation code tells the ALU what operation to perform and the operands used. It has 2 storage components named ACCUMULATOR register and AUXILIARY B register

The basic operations of ALU are

- ARITHMETIC OPERATIONS like addition, subtraction, multiplication and division
- LOGICAL OPERATIONS like and, or, not, nand, nor & xor. All are bitwise operations.
- COMPARISON OPERATIONS like greater than, lesser than, and equals to.
- SHIFTING AND ROTATE OPERATIONS like left shifting, left rotation, right shifting, and right rotation.
- CODE CONVERSION, converting Binary to Gray, XS-3, XS-5, and BCD. All operations are performed on ACCUMULATOR and AUXILIARY registers.

CONTENTS

1. INTRODUCTION

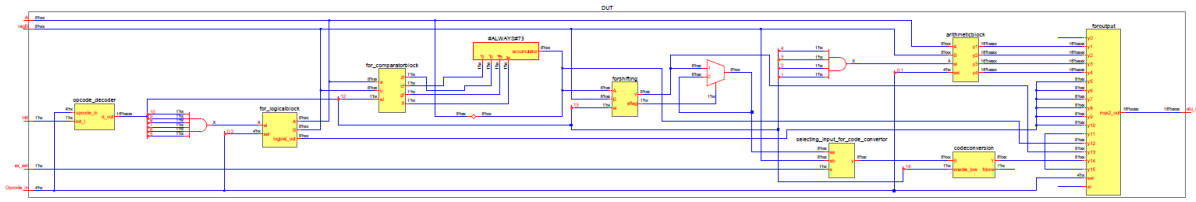
An arithmetic-logic unit is the part of a central processing unit that carries out arithmetic and logic operations on the operands(8-bit width) in computer instruction words(opcodes). The operation code tells the ALU what operation to perform and the operands used. It has 2 storage components named ACCUMULATOR register and AUXILIARY B register

The basic operations of ALU are given below

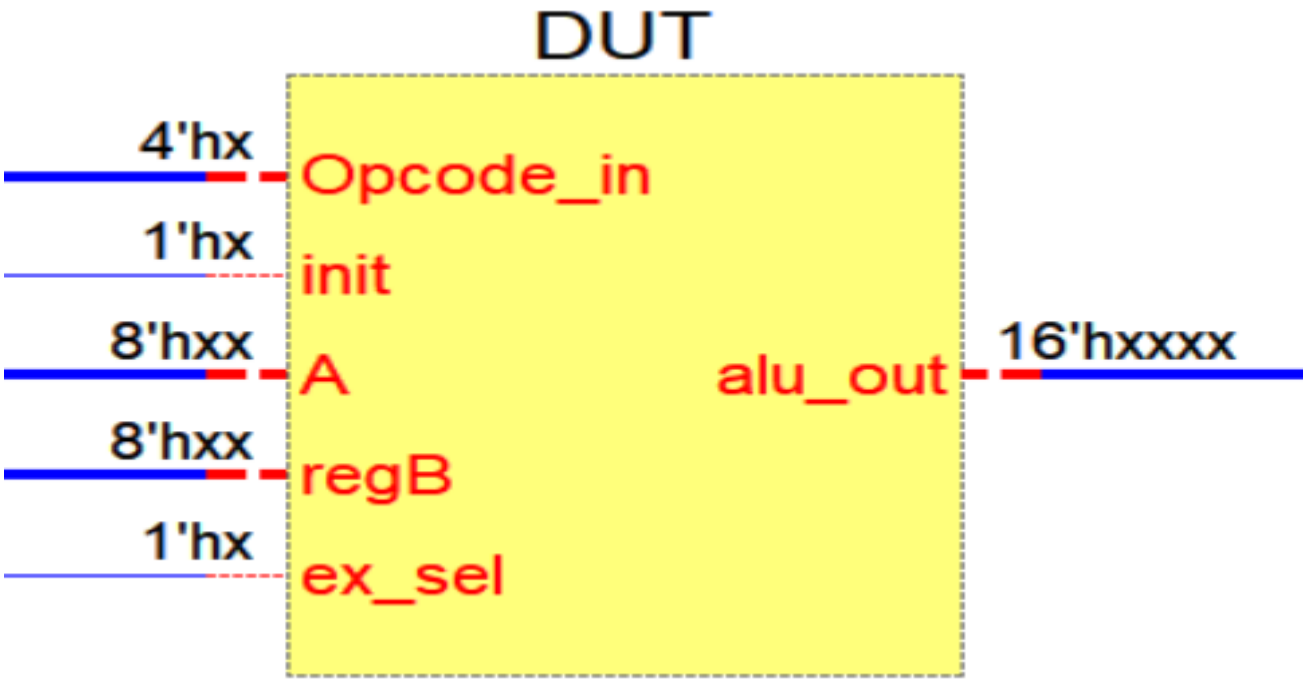
OPCODE	OPERATION
0001	Addition
0010	Subtraction
0011	Multiplication
0100	Division
0101	Bitwise AND
0110	Bitwise OR
0111	Bitwise NOT
1000	Bitwise NAND
1001	Bitwise NOR
1010	Bitwise XOR
1011	REFRESH

1100	8-bit Comparator
1101	Shifter/rotator
1110	Code Conversion
1111	Status

2. Top Module:



To view design click_here



Port list	Type	Size	From/To
Opcode_in	input	4	From user
init	input	1	From user
ex_sel	input	1	From user
A	Input	8	From user
regB	Input	8	From user
alu_out	output	16	To user

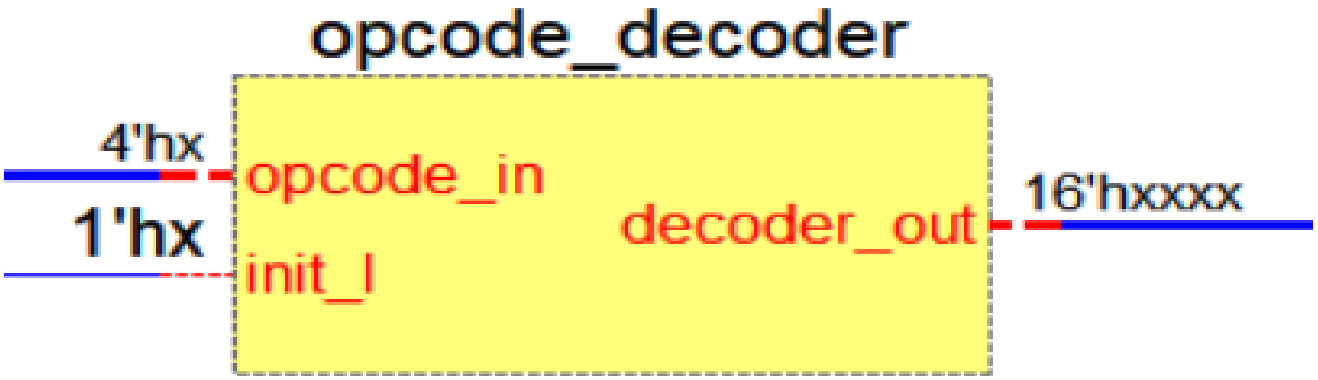
Top module have input operands A(accumulator) and B auxiliary register with size=8bit

Opcode_in tells what operation needs to be performed on operands

ex_sel is for selecting input for code conversion block.

init acts like active low reset

3. DECODER BLOCK:



Port list	Type	Size	From/To
opcode	input	4	From user

init_l	input	1	From user
d_out	output	16	To all blocks

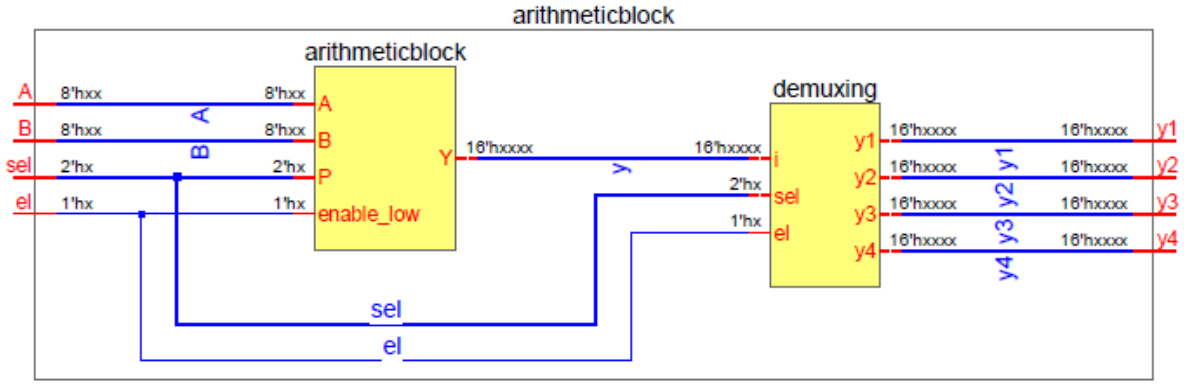
A decoder is a multiple-input, multiple-output combinational logic circuit. It converts the n-bit data inputs into the coded 2^n outputs. It decodes the information hidden by the encoder. The input of the active low decoder is opcode (Bitsize=4), While decoding, it places logic 0 at one of its outputs to create the exact code.

For a different combination of n-bit binary inputs, it produces a 2^n coded output. The coded output is produced based on each binary input. The decoder block consists of one enable input init which is active low and outputs of decoder blocks which act as enable input (active low)for other blocks in the alu. This block consists of 4 to 16 decoder which is controlled by a 4-bit opcode according to the opcode it will select the operation as follows

OPCODE	OPERATION
0001	Addition
0010	Subtraction
0011	Multiplication
0100	Division
0101	Bitwise AND
0110	Bitwise OR
0111	Bitwise NOT
1000	Bitwise NAND
1001	Bitwise NOR

1010	Bitwise XOR
1011	REFRESH
1100	8-bit Comparator
1101	Shifter/rotator
1110	Code Conversion
1111	Status

4. ARITHMETIC BLOCK:



Port list	Type	Size	From/To
A	Input	8	From user
B	Input	8	From user
sel	Input	2	From user
el	Input	1	From decoder output
y1	Output	16	To result fetching block
y2	Output	16	To result fetching block
y3	Output	16	To result fetching block
y4	Output	16	To result fetching block

This block consists of four operations:

The inputs of this block are accumulator(A) and auxiliary B register which of 8-bit size and outputs of operation will be stored in 16 bit out_arthmetic because in this block max size of the output of multiplication will be of 16 bit so the whole block of the output size will be the 16-bit size. According to the output of the decoder, the arithmetic block will be activated by active low enable input.sel input selects which operation needs to be performed

4.1 Addition:

When opcode 0001 is selected the addition operation will get activated, it will add two 8-bit numbers, and the result of sum and carry will be stored in a 16-bit Y register.

4.2 Subtraction:

When opcode 0010 is selected the subtraction operation will get activated; it will subtract two 8-bit numbers and the result will be stored in a 16-bit Y register.

4.3 Multiplication:

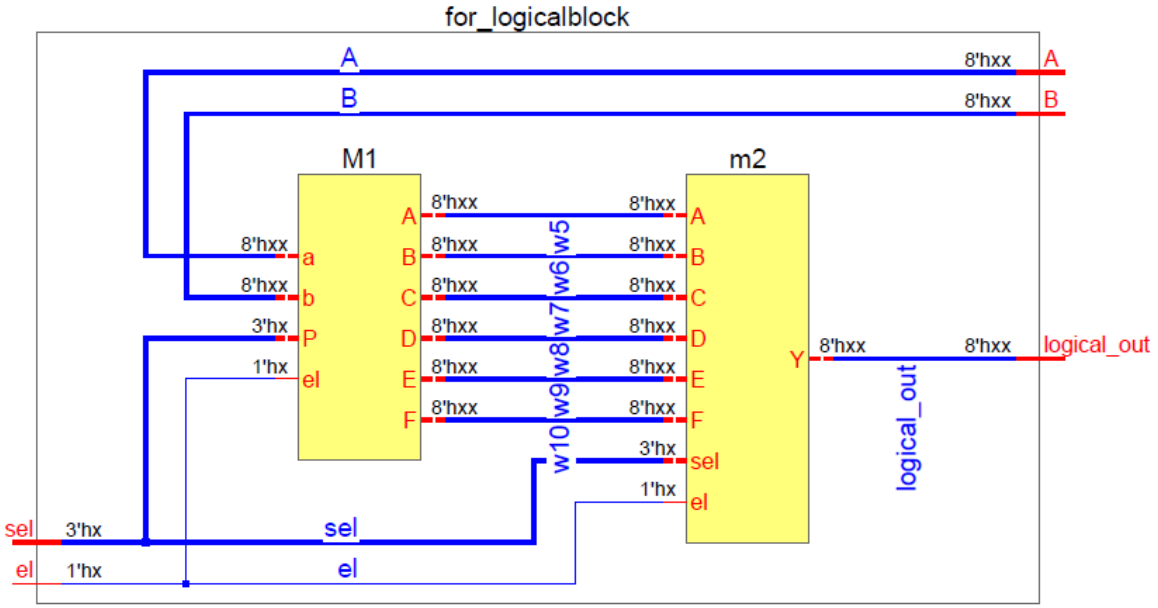
when opcode 0011 is selected the multiplication process will get activated. It will multiply two 8-bit numbers and the result will be stored in a 16-bit Y register.

4.4 Division:

When opcode 0100 is selected then the division process will get activated and it will divide two 8-bit numbers and the result will be stored in a 16-bit Y register.

The output of arithmetic operations is stored in a 16-bit Y register and given to a demultiplexer(1x4). The demux takes Y register data and based on the selection line(opcode) it will give output(y1,y2,y3,y4) to the result fetching block

5. Logical block



Port list	Type	Size	From/To
A	Input	8	From user
B	Input	8	From user
sel	Input	3	From user
el	Input	1	From decoder output
logical_out	Output	8	To result fetching block

This block consists of 6 operations:

The inputs of this block are accumulator A and auxiliary B register which of 8-bit size and outputs of the operation will be stored in 8-bit output (logical_out).

5.1 AND operation:

When opcode 0101 is selected then AND operation will be done on both inputs of 8-bit size and the result will be stored in register A which is of 8-bit size.

5.2 OR operation:

When opcode 0110 is selected then OR operation will be done on both inputs of 8-bit size and the result will be stored in register B which is of 8-bit size.

5.3 NOT operation:

When opcode 0111 is selected then NOT operation will be done on input A of 8-bit size and the result will be stored in register C which is of 8-bit size.

5.4 NAND operation:

When opcode 1000 is selected then NAND operation will be done on both inputs of 8-bit size and the result will be stored in register D which is of 8-bit size.

5.5 NOR operation:

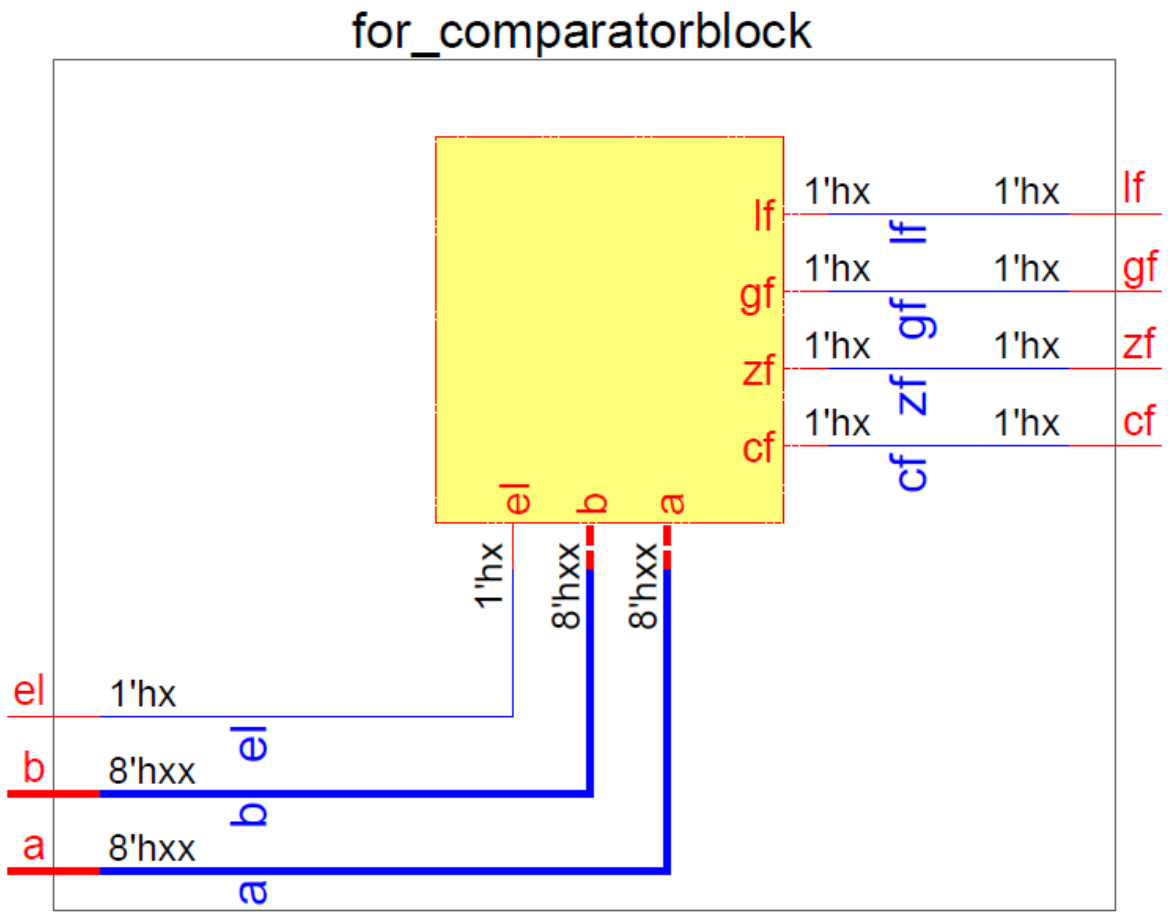
When opcode 1001 is selected then NORoperation will be done on both inputs of 8-bit size and the result will be stored in register E which is of 8-bit size.

5.6 XOR operation:

When opcode 1010 is selected then XOR operation will be done on both inputs of 8-bit size and the result will be stored in register F which is of 8-bit size.

The above operations will get activated by enabling the active low input from the decoder block then the output of operations stored in A, B, C, D, E, and F registers(8-bit) and given to mux to get single output from logical block(logical_out) a then logical_out given to the result fetching block.

6. COMPARATOR BLOCK:

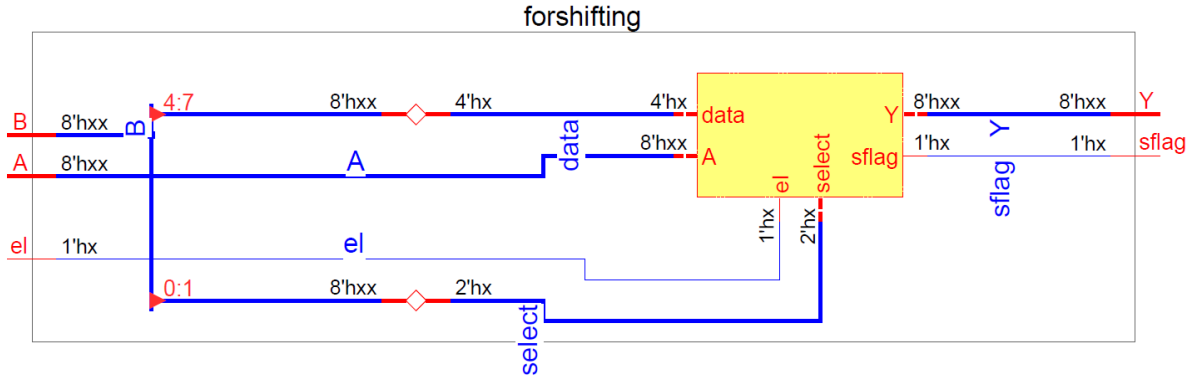


Port list	Type	Size	From/To
a	Input	8	From user
b	Input	8	From user
el	Input	1	From decoder output
gf	Output	8	To result fetching block
lf	Output	1	To result fetching block
zf	Output	1	To result fetching block
cf	Output	1	To result fetching block

The functionality of this block is which compares the no of bits input and produces output whether it is greater or equal or less. This block consists of two inputs accumulator (a) and auxiliary b register which is of 8-bit size and compares both 8-bit input data according to input it will produce output by checking the three conditions after completion of checking conditions the result will be stored in the output flags lf,gf,zf, and cf.

When opcode 1100 is selected, the comparator operation will get activated by providing the output of the decoder to enable input to the comparator block which is active low input.

7. SHIFTING BLOCK: Shifter/Rotator

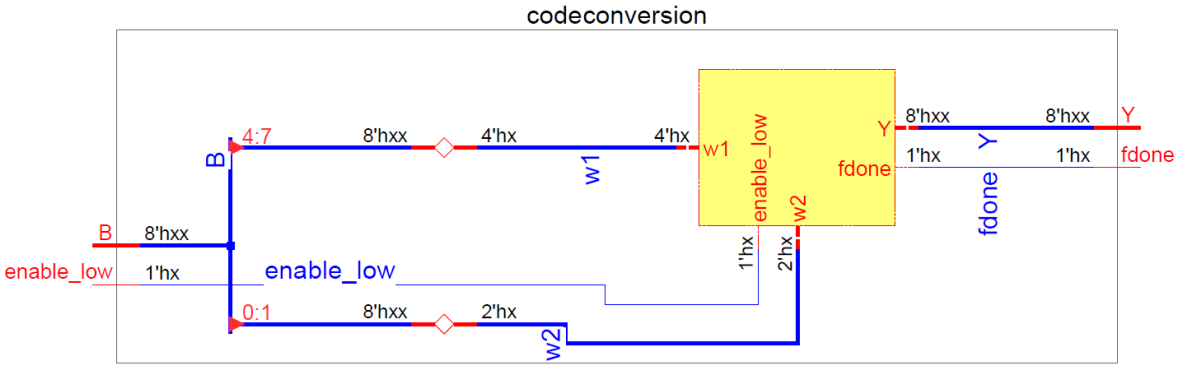


Port list	Type	Size	From/To
A	Input	8	From user
B	Input	8	From user
el	Input	1	From decoder output
sflag	output	1	To result fetching block
Y	output	8	To result fetching block

In this block shifting operation will be activated by enabling the output of the decoder to enable input to the shifting which is active low. it is used to shift data according to a given operation by using operators' left shift(<<) and right shift(>>). When opcode 1101 is selected shifting block operation gets activated by providing an output of the decoder as enable input to shift block. It has two 8 bits size Inputs accumulator(A) an auxiliary b register (B) and two outputs. output Y is of size 8 bit and sflag is of size 1 bit.

As they accept only one 8-bit operand which is the auxiliary b register from the lower bits of the b register b[3:0] decides the type of shifting or rotations and the higher bits b[7:4] will decide the no of shifting or rotating as per the command which will perform on A bit operand.

8. CODE CONVERSION BLOCK:



Port list	Type	Size	From/To
B	Input	8	From user
enable_low	Input	1	From decoder output
Y	output	8	To result fetching block
fdone	output	1	To result fetching block

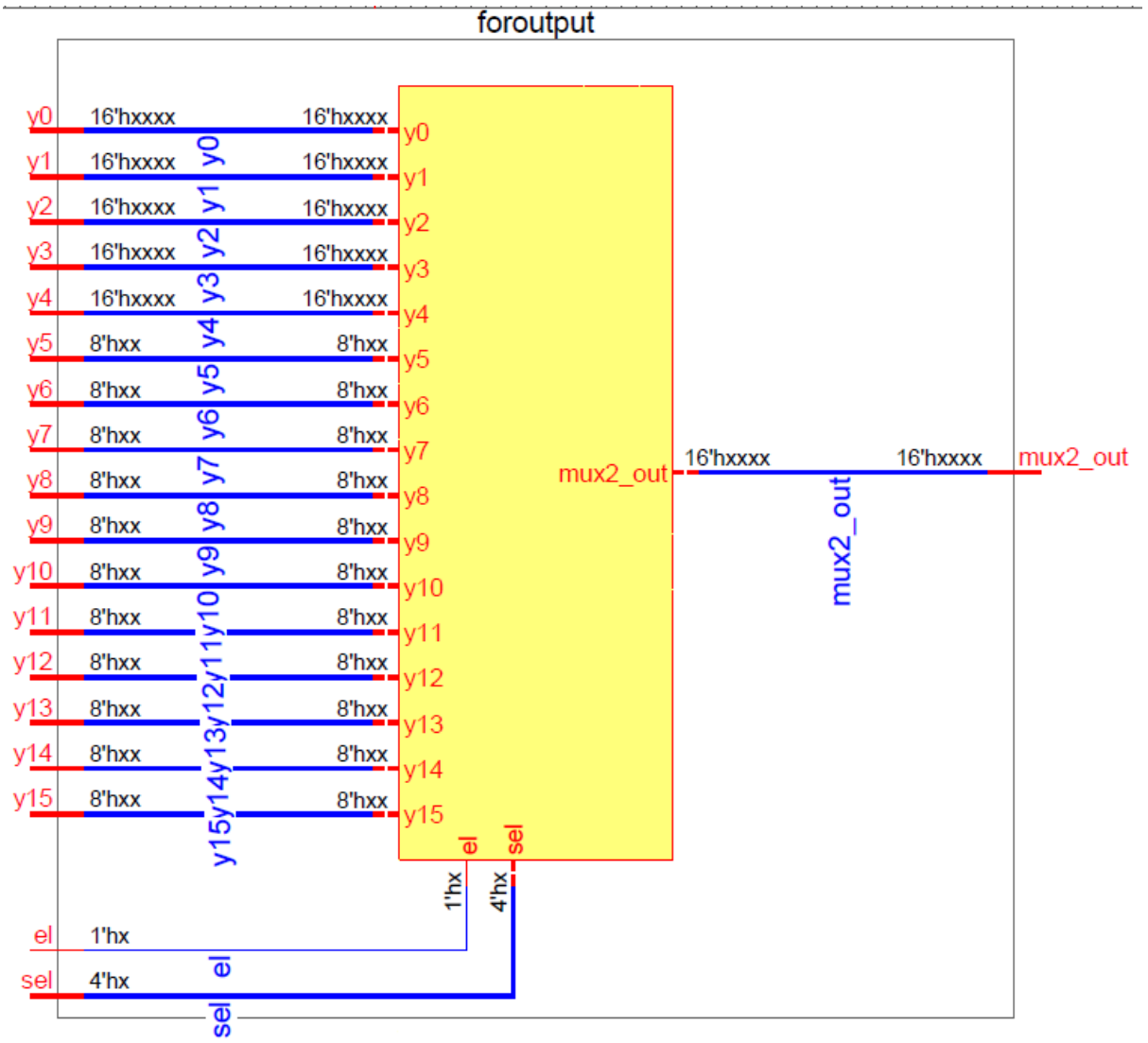
A code converter is a logic circuit that converts one binary coded to another type of code depends on the selection.When opcode 1110 is selected code conversion operation will get selected by enabling the output decoder to enable input to the code conversion block as an active low input.

In this block, it consists of a 4bit binary value from the auxiliary b register as input the LSB[3:0] selects the type of conversion and MSB[7:4] provides the binary values to decide a number of times to do the operation.

The code conversion needs to perform as per the opcode as follows:

Selection	Type of conversion
00	Binary to BCD
01	Binary to Gray code
10	Binary to XS-3
11	Binary to XS-5

RESULT FETCHING BLOCK



Port list	Type	Size	From/To
y0	Input	16	zzzz(high impedance)
y1	Input	16	From arithmetic block
y2	Input	16	From arithmetic block

y3	Input	16	From arithmetic block
y4	Input	16	From arithmetic block
y5	Input	8	From logical block
y6	Input	8	From logical block
y7	Input	8	From logical block
y8	Input	8	From logical block
y9	Input	8	From logical block
y10	Input	8	From logical block
y11	Input	8	zzzz(high impedance)
y12	Input	8	From comparator block
y13	Input	8	From shifter block
y14	Input	8	From code converter block
y15	Input	8	zzzz(high impedance)
mux2_out	output	16	To user

Result fetching block takes the input from the all others block results and based on opcode it will give the only one output at a time(size=16-bit).