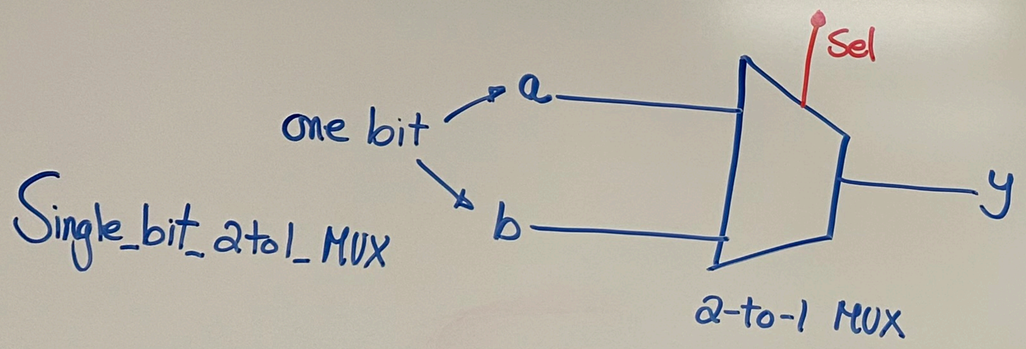
Design # 2

Part (1)



Single_bit_2to1_MUX

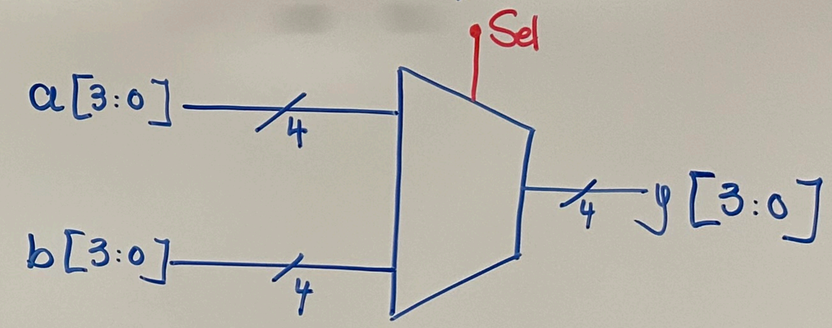
(1) Create a truth table

(2) Build the K-map

(3) Find the switching function SOP for y

(4) Write the Verilog for your design & test the syntax

Part (2)



Use the design in part (1) to build a 4bit 2-to-1 MUX

Four_bit_2to1_MUX

Truth Table for Single_bit_Mux:

A	B	Sel	y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

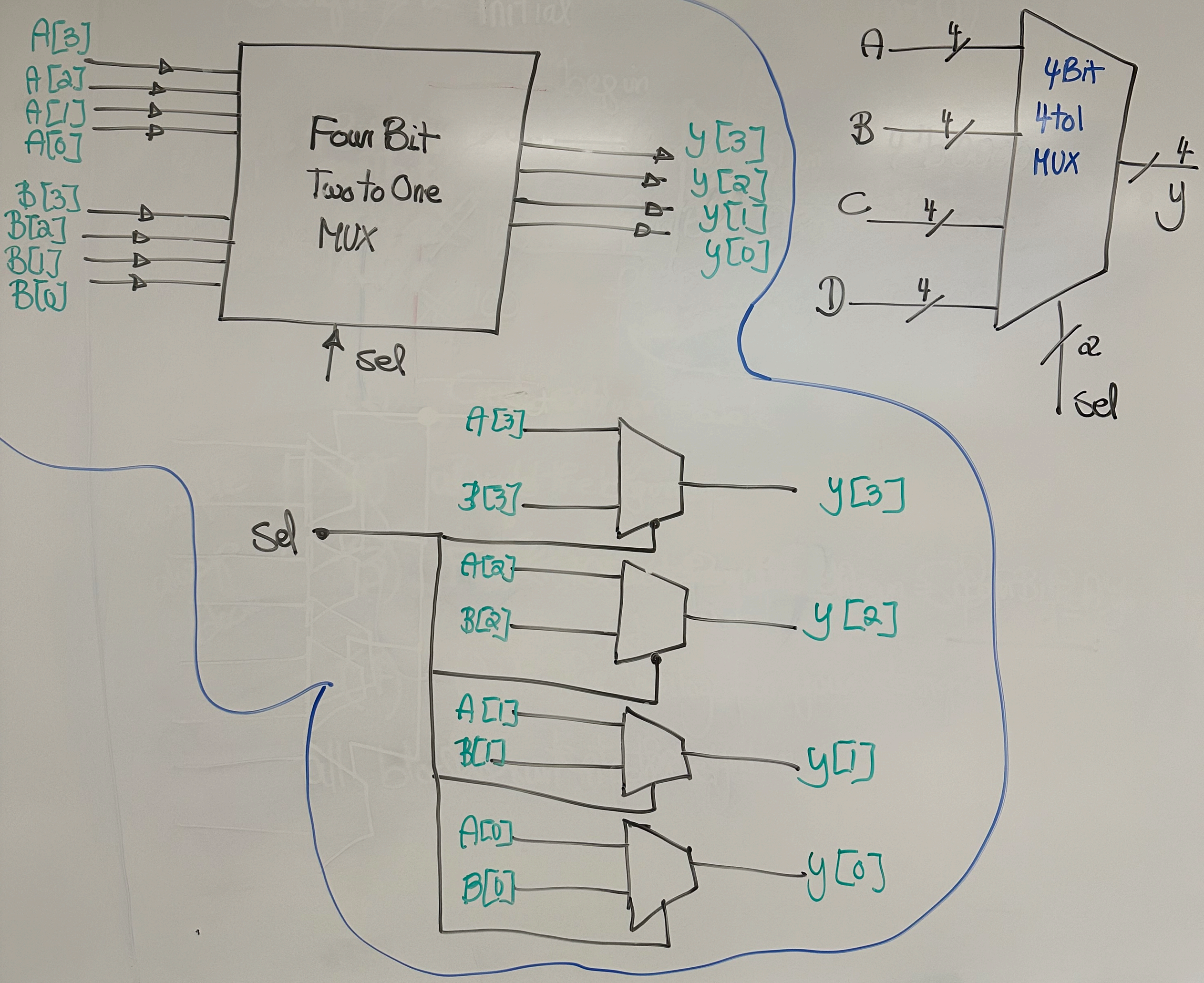
$A\bar{S}el$

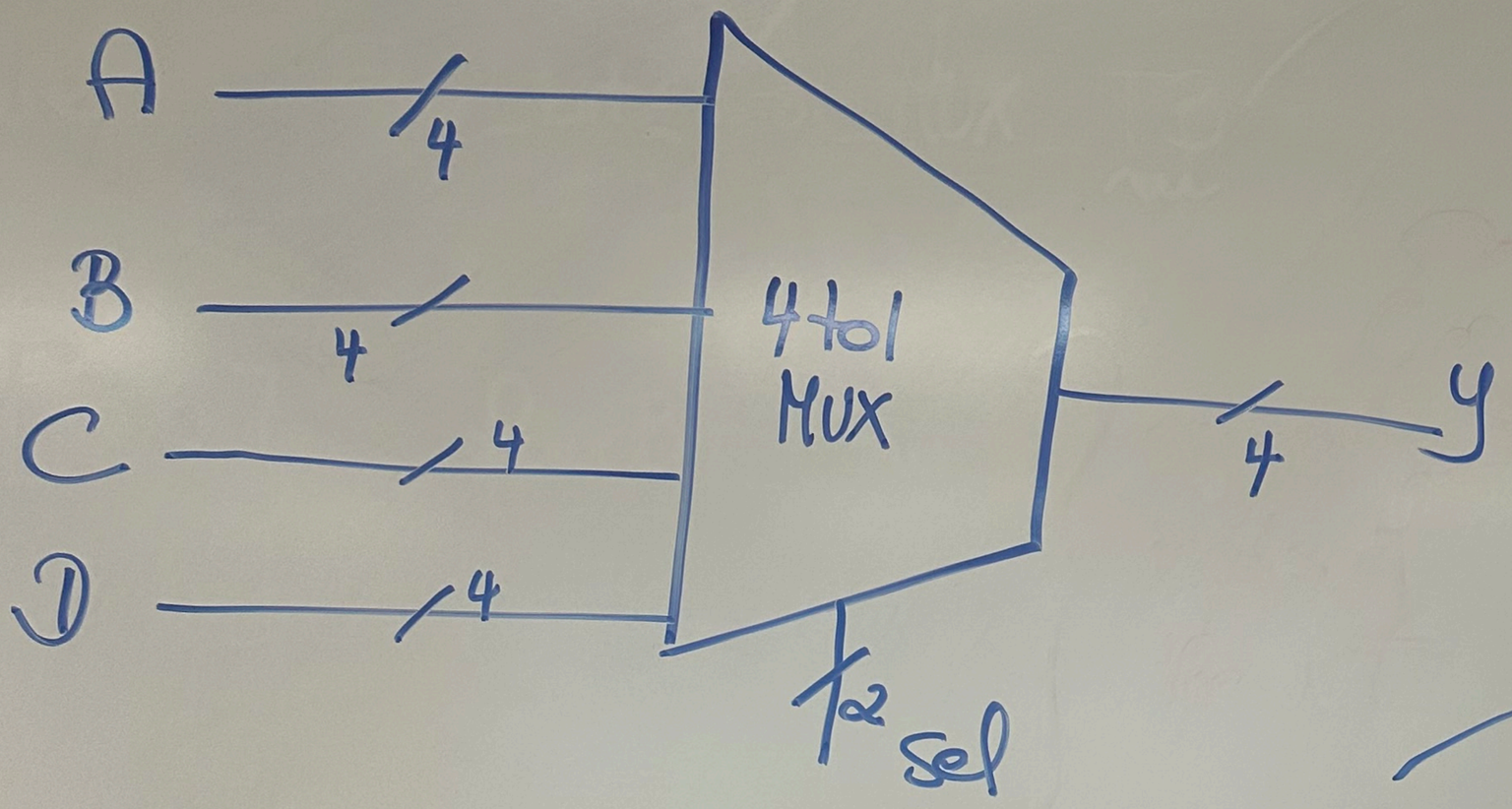
$+ BSel$

K-map

Sel \ AB	00	01	11	10
0	0	0	1	1
1	0	1	1	0

part II


A[3]
A[2]
A[1]
A[0]
B[3]
B[2]
B[1]
B[0]
Four Bit
Two to One
MUX
sel
y[3]
y[2]
y[1]
y[0]
A
B
C
D
4
4Bit
4to1
MUX
4
y
2
sel
A[3]
B[3]
sel
y[3]
A[2]
B[2]
y[2]
A[1]
B[1]
y[1]
A[0]
B[0]
y[0]



Sel	y
00	A
01	B
10	C
11	D

Truth table
for the 4 to 1 Mux

// Testbench module

name of your module to be tested

TB for test bench

module Four_bit_2to1_MUX_TB ;

notice that the test bench module does not have any ports. (simulation)

reg [3:0] fourBit_A_in ;
reg [3:0] fourBit_B_in ;
reg Sel_in ;

{reg} original inputs of the UUT

wire [3:0] four Bit_Y_out ;

original output of the UUT

{wires} ports

Four_bit_2to1_MUX UUT (fourBit_Y_out , fourBit_A_in , four Bit_B_in , sel_in) ;

Four_bit_2to1_MUX → instantiation of the module under test

UUT → Instance name

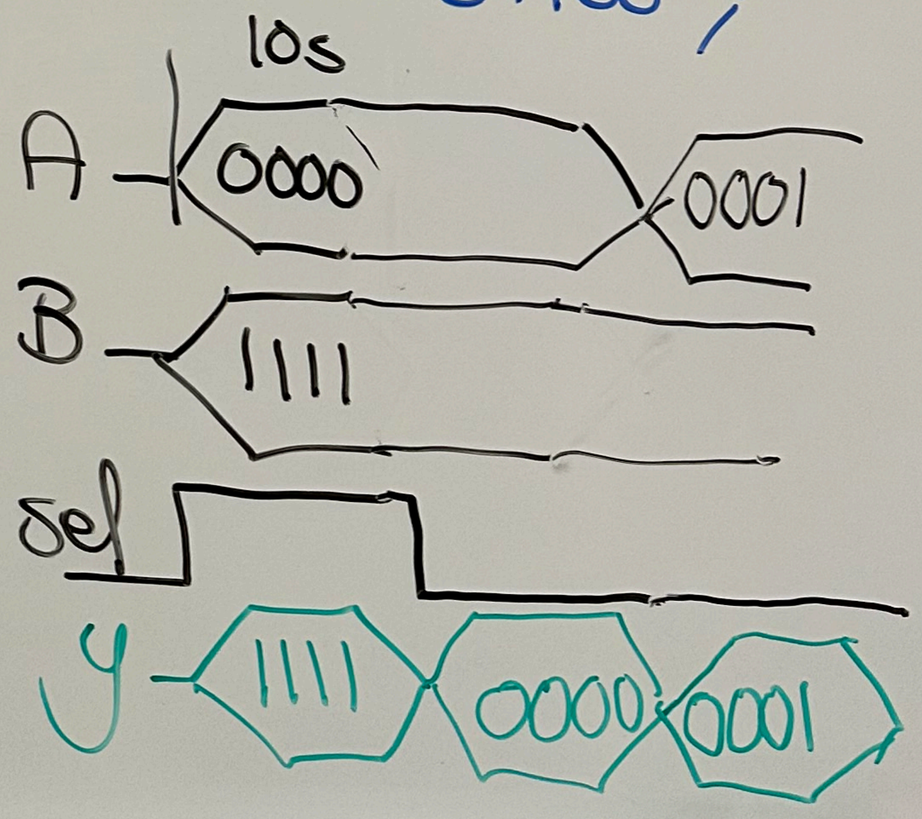
Unit Under Test

```
initial
  begin
    fourBit_A_in = 4'b0000 ;
    fourBit_B_in = 4'b1111 ;
    sel_in = 1 ;
#10 sel_in = 0 ;
#10 fourBit_A_in = 4'b0001 ;
#10 fourBit_A_in = 4'b0010 ;
#10 fourBit_A_in = 4'b1100 ;
    .
    .
    .
  end
endmodule
```

Time units

Bit_A_in + 1 ;

```
wire
reg
reg
reg
Four
inst
modul
```



Example initial block

```
initial
  begin
    fourBit_A_in = 4'b0000;
    #100 fourBit_A_in = 4'b0001;
    #100 fourBit_A_in = 4'b0010;
    :
  end
```

example always block

```
always begin
  #100 fourBit_A_in = fourBit_A_in + 1;
end
```

all blocks run in parallel