Bangladesh University of Engineering and Technology

Department of Electrical and Electronics Engineering

Course Number: EEE 304

Course Title: Digital Electronics Laboratory

Experiment Number: 03

Name of the Experiment(s):

Design, Simulation, and Implementation of Combinational Circuits: Decoder/Encoder/Multiplexer Circuit using 74 series ICs and Verilog HDL

Prepared by: Tasmin Khan

Student ID: 1906044

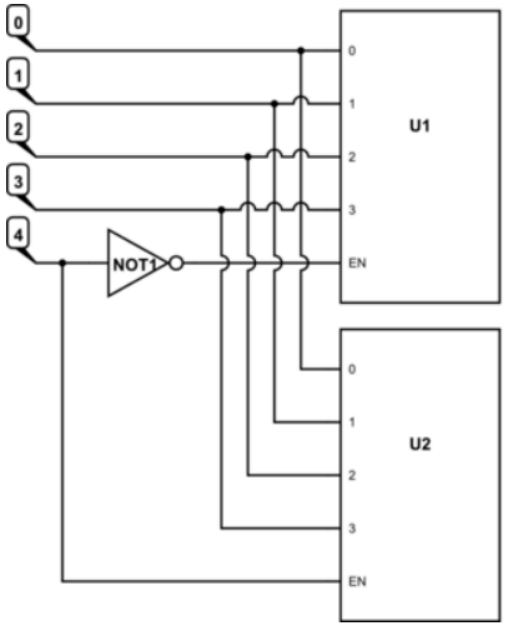
Level:3 Term:2

Report Task 1

Question: Construct a 5 bit (i.e 5 line to 32 line) decoder circuit by cascading two 4 line to 16 line decoder.

5 bit to 31 line Decoder with two 4 bit to 16 line Decoder

Circuit Schematic:



Here the MSB bit A4 is directing which decoder will be working being the enable bit.

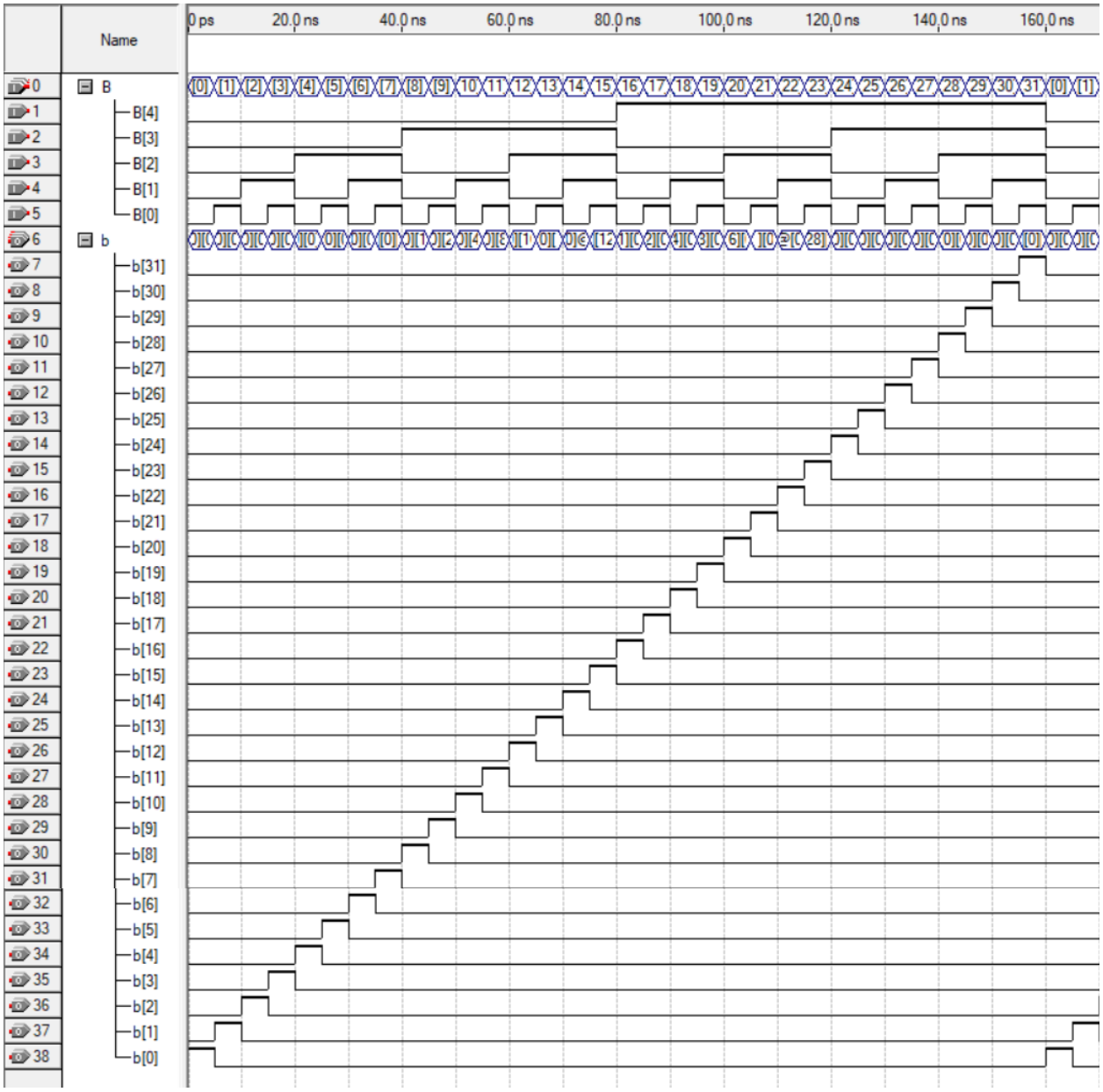
For A4=0, the first decoder U1 works for a numerical range of 0-15

For A4=1, the second decoder U2 works for a numerical range of 16-32

Verilog code:

```
1  module decoder(B,b);
2  input [4:0]B;
3  output [31:0]b;
4  wire enb;
5  wire [3:0]AA;
6  assign enb=B[4];
7  assign AA[3:0]=B[3:0];
8  assign b[15:0] = (a((!enb),AA[3:0]));
9  assign b[31:16] = (a((enb),AA[3:0]));
10
11 function automatic [15:0]a;
12 input enb;
13 input [3:0]A;
14 begin
15 a[0]=(enb)&(!A[3])&(!A[2])&(!A[1])&(!A[0]);
16 a[1]=(enb)&(!A[3])&(!A[2])&(!A[1])&(A[0]);
17 a[2]=(enb)&(!A[3])&(!A[2])&(A[1])&(!A[0]);
18 a[3]=(enb)&(!A[3])&(!A[2])&(A[1])&(A[0]);
19 a[4]=(enb)&(!A[3])&(A[2])&(!A[1])&(!A[0]);
20 a[5]=(enb)&(!A[3])&(A[2])&(!A[1])&(A[0]);
21 a[6]=(enb)&(!A[3])&(A[2])&(A[1])&(!A[0]);
22 a[7]=(enb)&(!A[3])&(A[2])&(A[1])&(A[0]);
23 a[8]=(enb)&(A[3])&(!A[2])&(!A[1])&(!A[0]);
24 a[9]=(enb)&(A[3])&(!A[2])&(!A[1])&(A[0]);
25 a[10]=(enb)&(A[3])&(!A[2])&(A[1])&(!A[0]);
26 a[11]=(enb)&(A[3])&(!A[2])&(A[1])&(A[0]);
27 a[12]=(enb)&(A[3])&(A[2])&(!A[1])&(!A[0]);
28 a[13]=(enb)&(A[3])&(A[2])&(!A[1])&(A[0]);
29 a[14]=(enb)&(A[3])&(A[2])&(A[1])&(!A[0]);
30 a[15]=(enb)&(A[3])&(A[2])&(A[1])&(A[0]);
31 end
32 endfunction
33
34 endmodule
```

Vector Waveform:

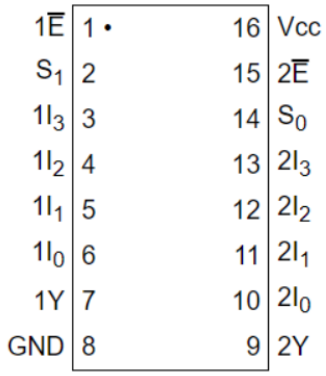


Report Task 2

Question: Implement an 8 line to 1 line MUX using two 4 line to 1 line MUX (found in IC 74153) and an OR gate.

8 line to 1 line MUX using two 4 line to 1 line MUX and OR gate

We will be using IC74153 for this task. The pin diagram is given in the picture. Here, 1E and 2E pins are 'enable' pins. We will use only one enable pin to join two such 4 line to 1 line multiplexers.

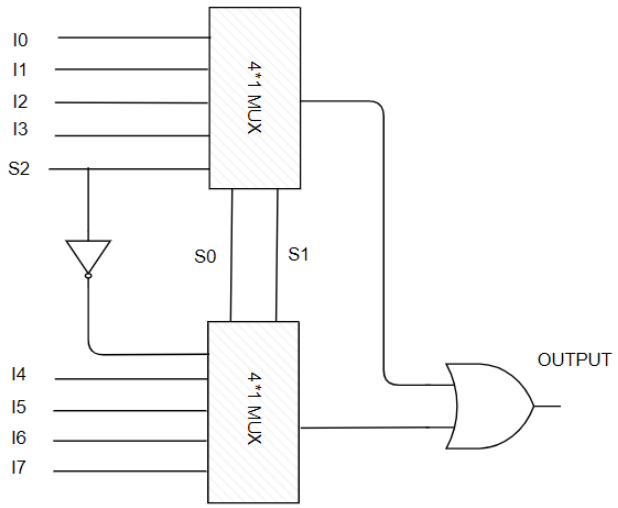


Truth table for 8 line to 1 line MUX:

Input			Output
S_2	S_1	S_0	I [10:0]
0	0	0	I_0
0	0	1	I_1
0	1	0	I_2
0	1	1	I_3
1	0	0	I_4
1	0	1	I_5
1	1	0	I_6
1	1	1	I_7

Here, S2 will be used as the enable pin.

Circuit schematic:

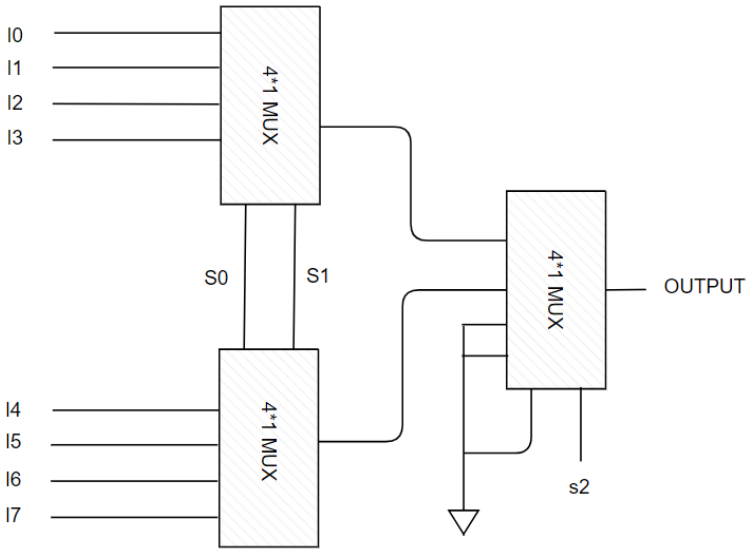


Report task 3

Question: Design the same circuit as described in report task 2 with three 4-1 MUX

8 line to 1 line MUX using three 4 line to 1 line MUX

Circuit schematic:



Report Task 4

Question: Implement a 10 line to 1 line MUX using two 4 line to 1 line MUX, one 2 line to 1 line MUX and a 3 input or gate. Can you suggest any alternative design solution?

10 line to 1 line MUX using two 4 line to 1 line MUX,

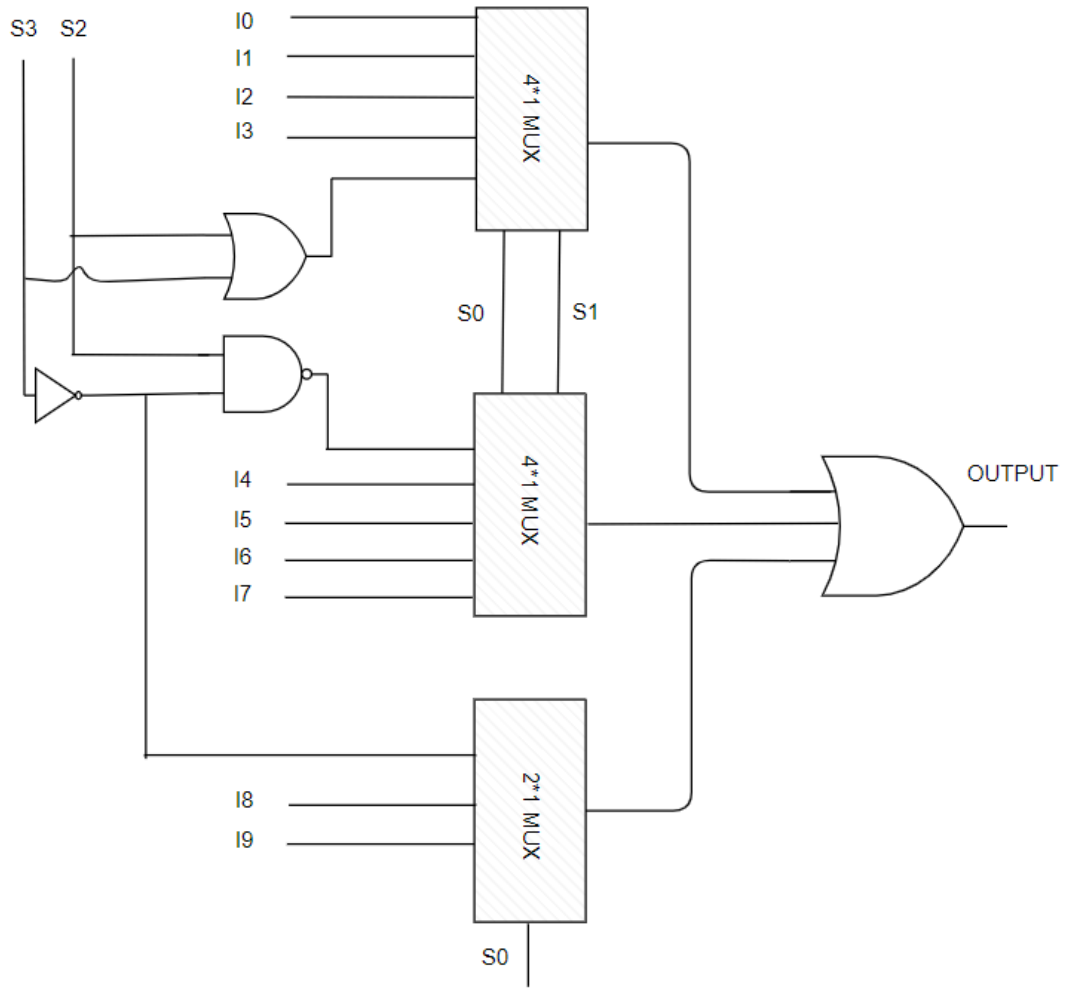
one 2 line to 1 line MUX and a 3 input OR gate

10 line to 1 line MUX truth table:

Input				**Output**
S_3	**S_2**	**S_1**	**S_0**	**I [10:0]**
0	0	0	0	I_0
0	0	0	1	I_1
0	0	1	0	I_2
0	0	1	1	I_3
0	1	0	0	I_4
0	1	0	1	I_5
0	1	1	0	I_6
0	1	1	1	I_7
1	0	0	0	I_8
1	0	0	1	I_9

Circuit diagram:

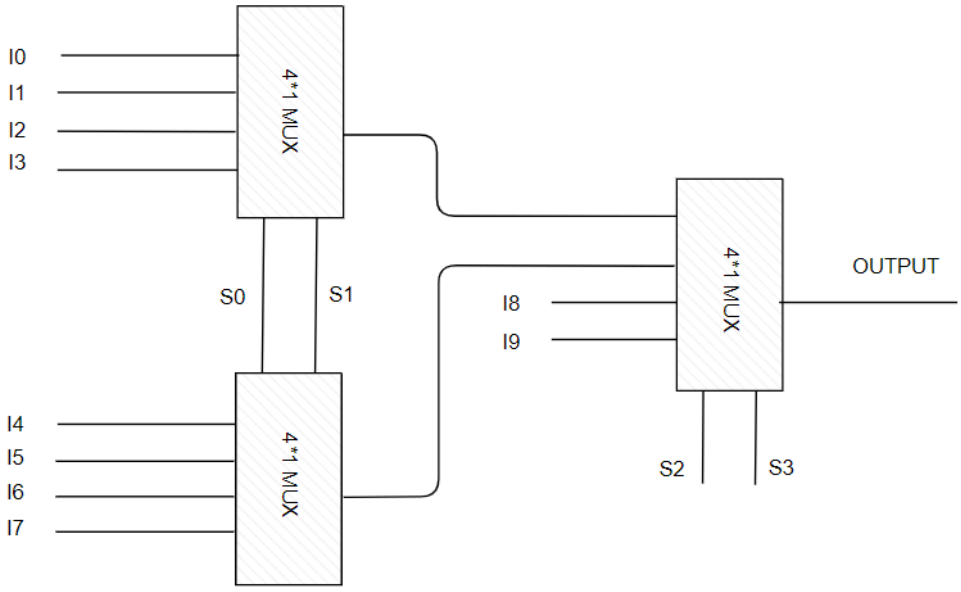
Here we will use S2 as the enable pin to differentiate between the two 4*1 MUX. WE will use one additional enable pin S3 to append another 2*1 MUX and thus we can construct a 10*1 MUX with the help of an OR gate at the final stage.



In physical IC, the enable pins are inverted automatically. So to implement an enable 1, we need to give input of enable 0. The enable pin connections in the circuit schematics were drawn keeping this in mind.

Alternative solution:

We can also implement the same circuit with just three 4*1 MUX as follows:



Report Task 5

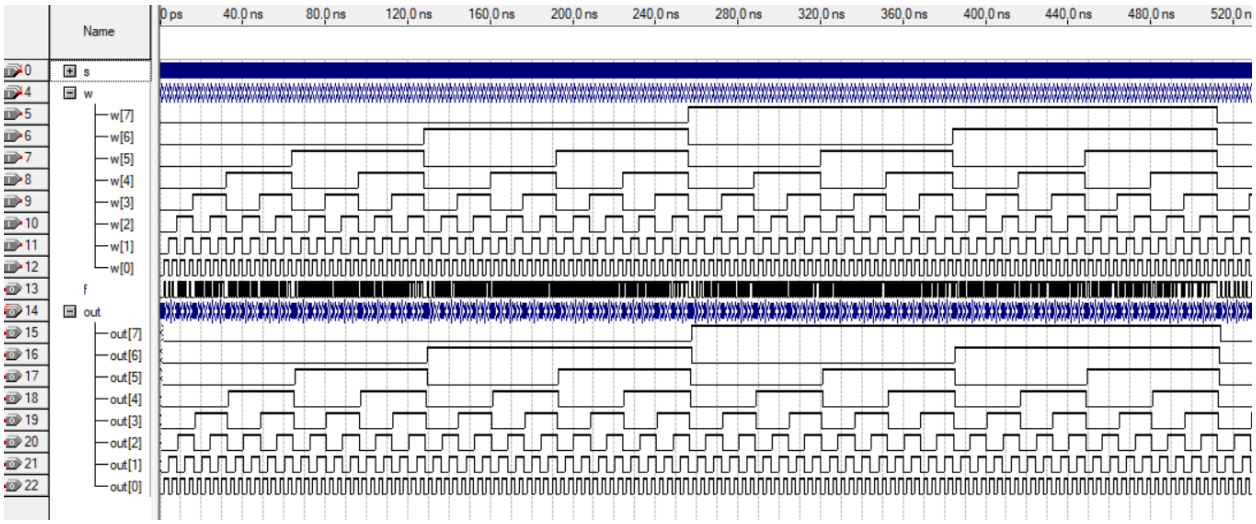
Question: Write the Verilog code of 8 input multiplexer-demultiplexer circuit. Simulate and test the circuit using Quratus II software. The timing diagram must be checked after Fitter and Assembler is run.

8 Input multiplexer-demultiplexer circuit:

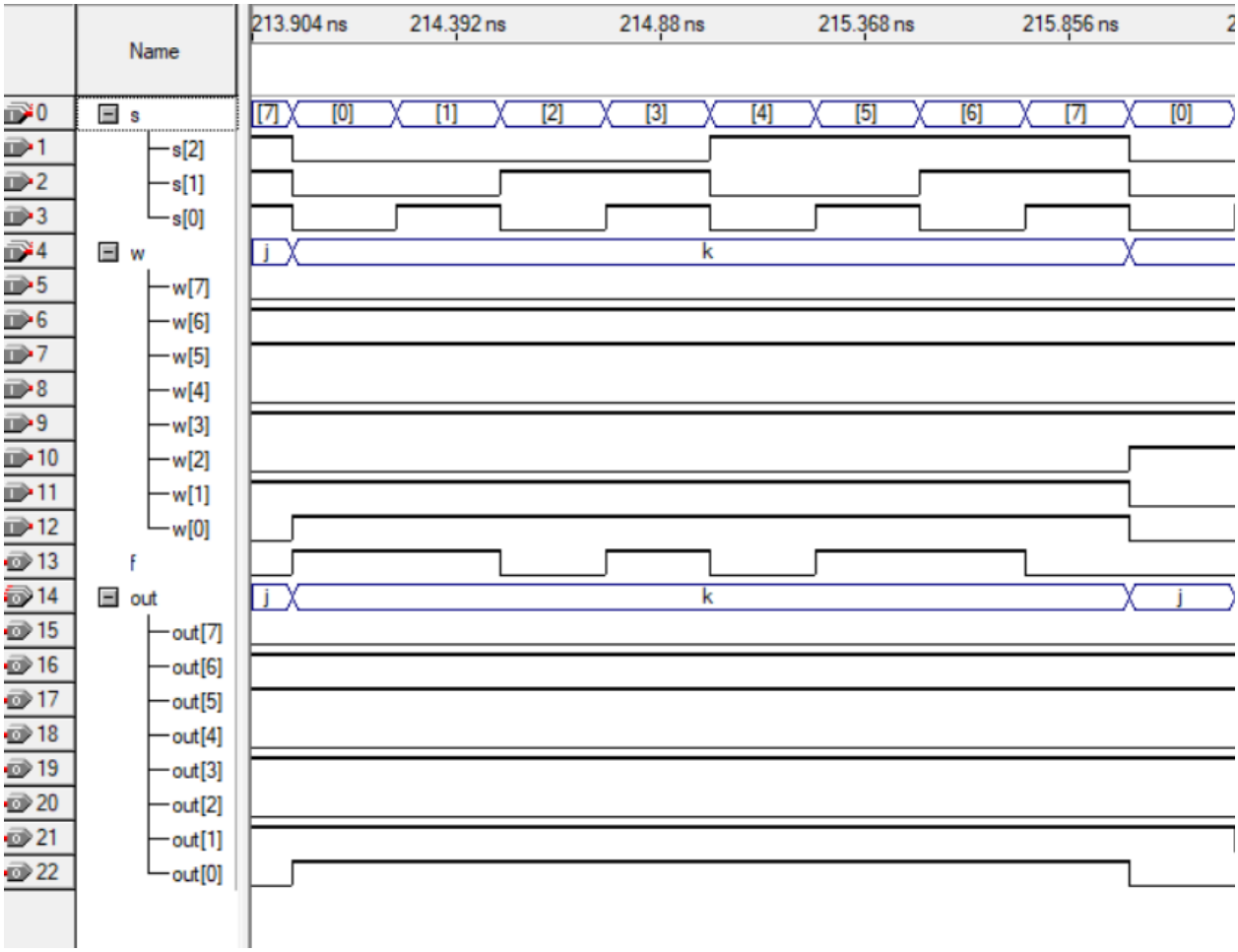
Verilog code:

```
1  module muxdemux(w,s,f,out);
2  input [7:0]w;
3  input [2:0]s;
4  output reg f;
5  output reg [7:0]out;
6
7  always @(w,s)
8  begin
9  if (s==0) f=w[0];
10 else if (s==1) f=w[1];
11 else if (s==2) f=w[2];
12 else if (s==3) f=w[3];
13 else if (s==4) f=w[4];
14 else if (s==5) f=w[5];
15 else if (s==6) f=w[6];
16 else if (s==7) f=w[7];
17 end
18
19 always @(f)
20 begin
21 if (s==0) out[0]=f;
22 else if (s==1) out[1]=f;
23 else if (s==2) out[2]=f;
24 else if (s==3) out[3]=f;
25 else if (s==4) out[4]=f;
26 else if (s==5) out[5]=f;
27 else if (s==6) out[6]=f;
28 else if (s==7) out[7]=f;
29 end
30
31 endmodule
32
```

Vector wave form:



Now we will see a zoomed in version for a full cycle of select input



The values from the zoomed in plot are listed below in a table:

s[2]	s[1]	s[0]	Input W	f	Output
0	0	0	w[0]=1	1	out[0]=1
0	0	1	w[1]=1	1	out[1]=1
0	1	0	w[2]=0	0	out[2]=0
0	1	1	w[3]=1	1	out[3]=1
1	0	0	w[4]=0	0	out[4]=0
1	0	1	w[5]=1	1	out[5]=1
1	1	0	w[6]=1	1	out[6]=1
1	1	1	w[7]=0	0	out[7]=0

Here, f is the output of the MUX block. Where it follows the input set by select inputs. The same select input works on the DeMUX block and sets the certain final output equal to f. Since we used the same select for both blocks, the input and output are identical in order.

Discussion:

Here, we implemented DeMUX, MUX and Decoder blocks using Verilog code or circuit schematic. We also learnt tricky parts like enable pins and select input. The simulated output matches with the theoretical explanation and so we can say this experiment was successful.