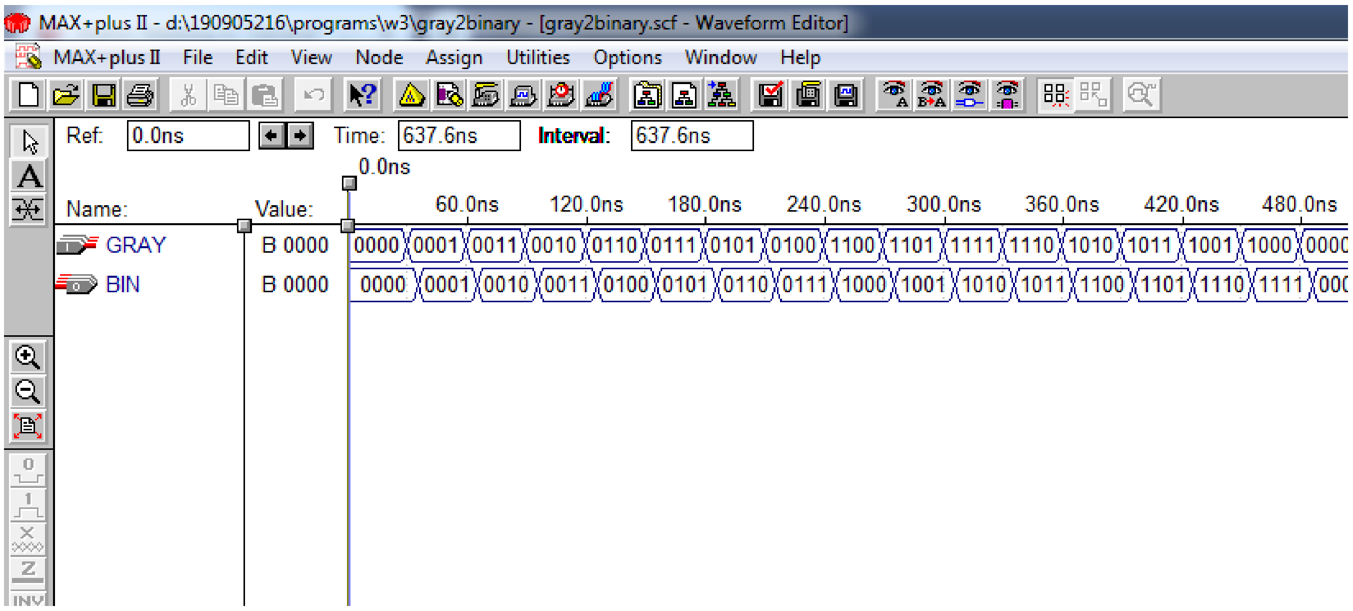
Week 3:

Q1. Gray Code to Binary Code

```
module gray2binary(GRAY, BIN);
        parameter n=4;
        input[n-1:0] GRAY;
        output [n-1:0] BIN;
        reg [n-1: 0] BIN;
        integer k;
        always @(GRAY)
        begin BIN[n-1] = GRAY[n-1];
        for(k = n-2; k>= 0; k = k-1)
        BIN[k] = BIN[k+1]^GRAY[k];
        end
endmodule
```

Wavefrom:

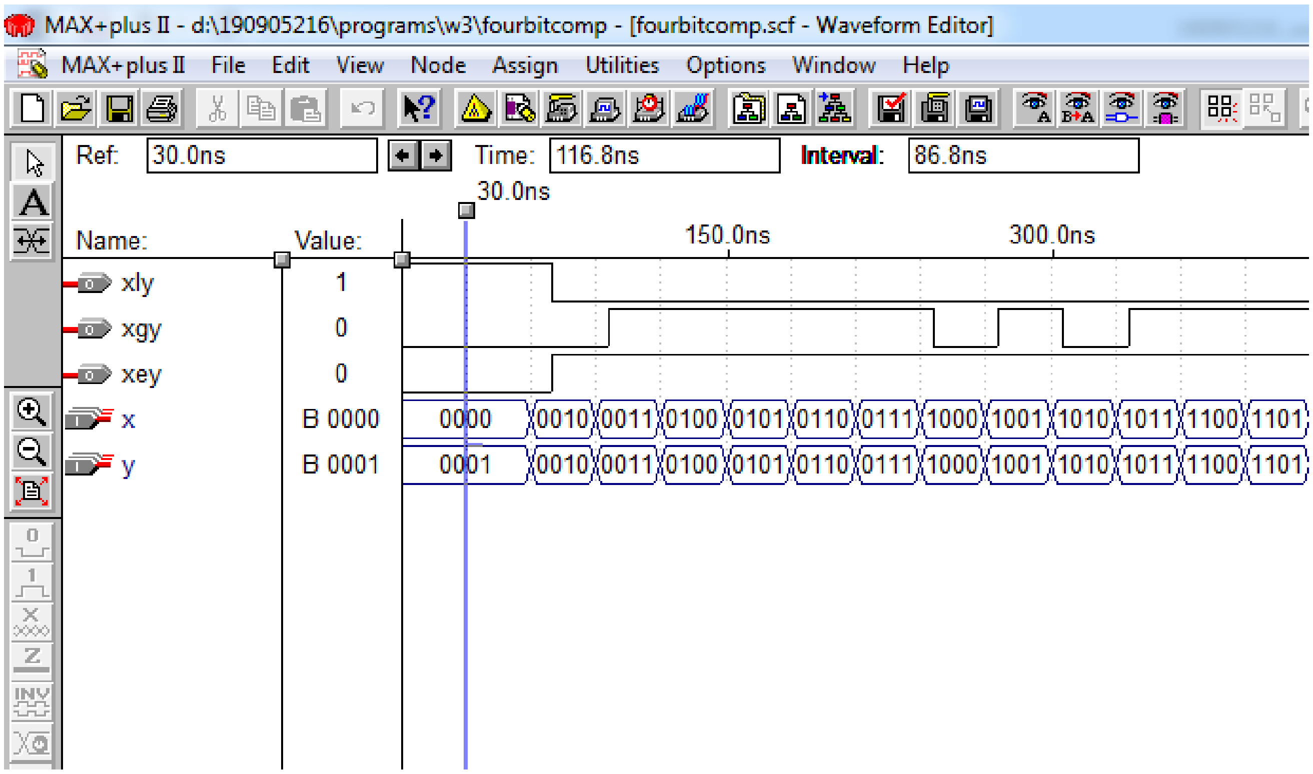

MAX+plus II - d:\190905216\programs\w3\gray2binary - [gray2binary.scf - Waveform Editor]
MAX+plus II
File
Edit
View
Node
Assign
Utilities
Options
Window
Help

```
Ref: 0.0ns    Time: 637.6ns    Interval: 637.6ns

Name:   Value:  0.0ns  60.0ns  120.0ns  180.0ns  240.0ns  300.0ns  360.0ns  420.0ns  480.0ns
GRAY    B 0000  0000 0001 0011 0010 0110 0111 0101 0100 1100 1101 1111 1110 1010 1011 1001 1000 0000
BIN     B 0000  0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111 000
```

Q2: 4 bit comparator using 2 bit comparator

```
module fourbitcomp(x,y,xly,xgy,xey);

input [3:0] x,y;

output xly,xgy,xey;

wire l1,l2,g1,g2,e1,e2;

twobitcomp stage0(x[0],y[0],x[1],y[1],l1,g1,e1);

twobitcomp stage1(x[2],y[2],x[3],y[3],l2,g2,e2);

assign xey = e1 & e2;

assign xgy = g2|(e2&g1);

assign xly = l2|(e2&l1);

endmodule


module twobitcomp(x0,y0,x1,y1,l,g,e);

input x0,y0,x1,y1;

output l,g,e;

wire i,j;

assign j = ~(x1^y1);

assign i = ~(x0^y0);

assign e = i&j;

assign g = (x1&~y1)|(j&x0&y0);

assign l = ~(g|e);

endmodule
```

Waveform for 4-bit comparator:



MAX+plus II - d:\190905216\programs\w3\fourbitcomp - [fourbitcomp.scf - Waveform Editor]

Ref: 30.0ns   Time: 116.8ns   Interval: 86.8ns

Name:	Value:
xly	1
xgy	0
xey	0
x	B 0000
y	B 0001

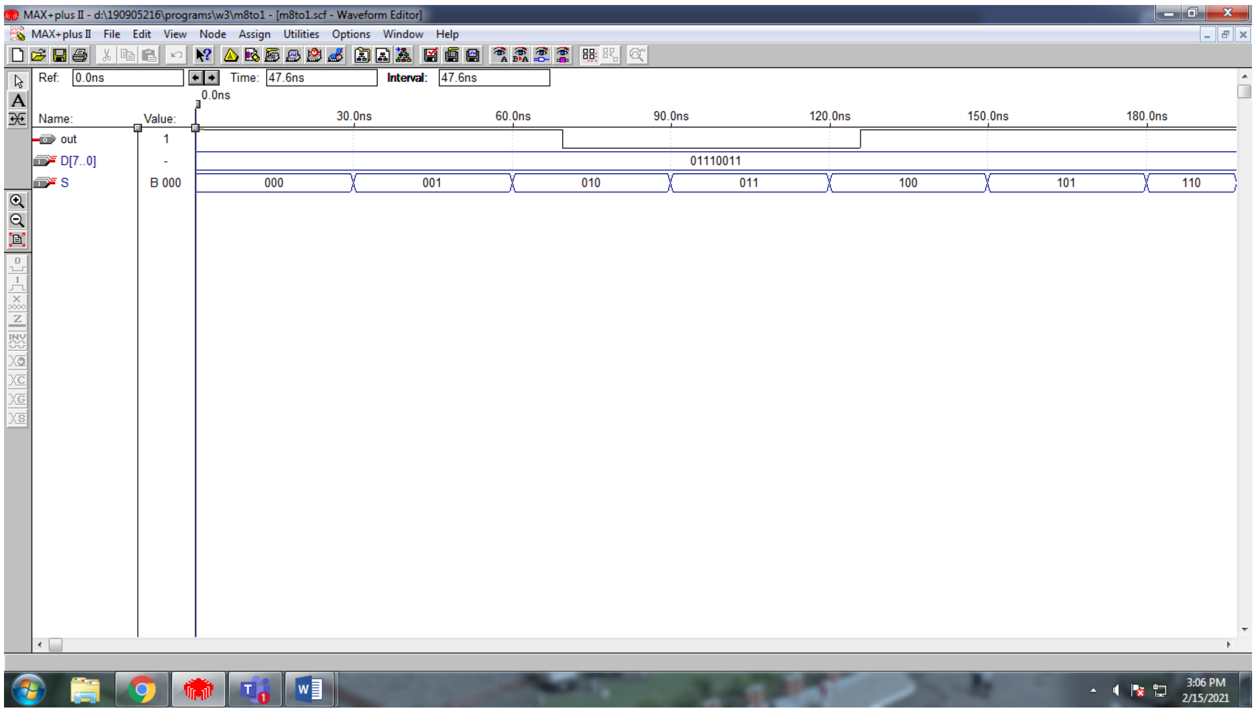
PTO->

Q3: Part 1

8 to 1 Mux:

```
module m8to1(D, S, out);
input [7:0] D;
input [2:0] S;
wire [7:0] D;
wire [2:0] S;
output out;
reg out;
always@(D or S)
begin
case(S)
0: out=D[0];
1: out=D[1];
2: out=D[2];
3: out=D[3];
4: out=D[4];
5: out=D[5];
6: out=D[6];
7: out=D[7];
endcase
end
endmodule
```

Waveform for 8 to 1 Mux:



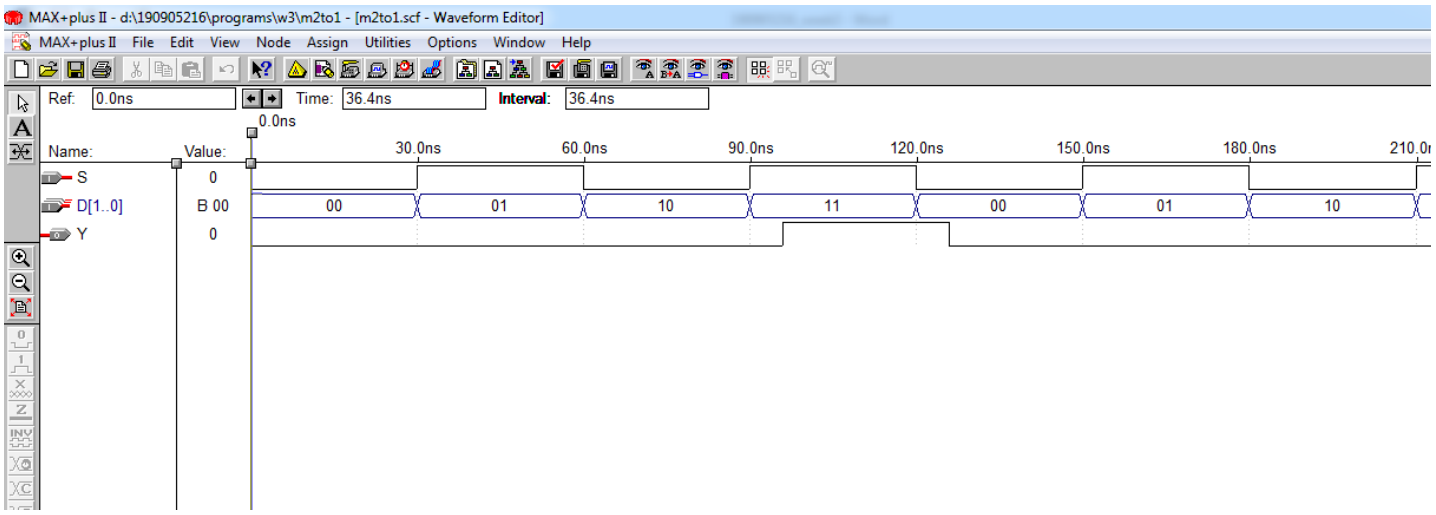
PTO->

Q3: Part 2

2 to 1 Mux:

```
module m2to1( D0, D1, S, Y);
input D0, D1, S;
wire D0, D1;
output Y;
reg Y;
always @(D0 or D1 or S)
begin
if(S)
Y= D1;
else
Y=D0;
end
endmodule
```

Waveform of 2 to 1 Mux:


MAX+plus II - d:\190905216\programs\w3\m2to1 - [m2to1.scf - Waveform Editor]
MAX+plus II File Edit View Node Assign Utilities Options Window Help

Ref: 0.0ns Time: 36.4ns Interval: 36.4ns

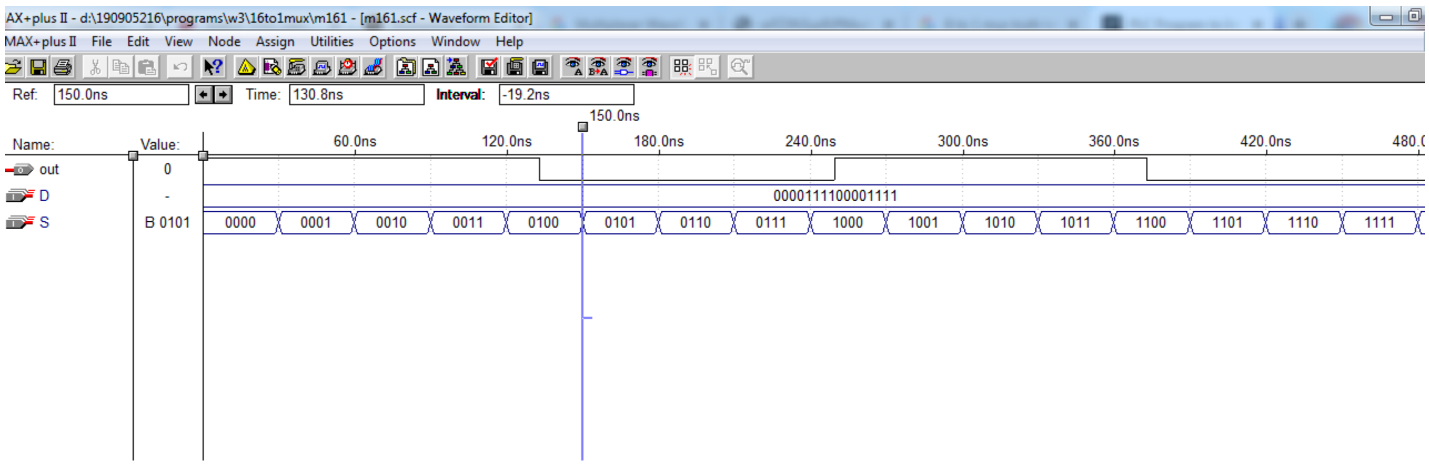
Name:	Value:	0.0ns	30.0ns	60.0ns	90.0ns	120.0ns	150.0ns	180.0ns	210.0
S	0								
D[1..0]	B 00	00	01	10	11	00	01	10	
Y	0								

16 to 1 Mux:

```
module m161(out, D, S);
input [15:0] D;
input [3:0] S;
output out;
reg out;
wire [15:0] D;
wire [3:0] S;
wire [1:0] x;
m81 m1(x[0], D[7:0], S[2:0]);
m81 m2(x[1], D[15:8], S[2:0]);
m21 m3(out, x[0], x[1], S[3]);
endmodule

module m21(out, D0, D1, S);
input D0, D1, S;
wire D0, D1;
output out;
reg out;
always @(D0 or D1 or S)
begin
if(S)
out= D1;
else
out=D0;
end
endmodule
```

```
module m81(out, D, S);

input [7:0] D;

input [2:0] S;

wire [7:0] D;

wire [2:0] S;

output out;

reg out;

always@(D or S)

begin

case(S)

0: out=D[0];

1: out=D[1];

2: out=D[2];

3: out=D[3];

4: out=D[4];

5: out=D[5];

6: out=D[6];

7: out=D[7];

endcase

end

endmodule
```

Waveform for a 16 to 1 Mux


AX+plus II - d:\190905216\programs\w3\16to1mux\m161 - [m161.scf - Waveform Editor]
MAX+plus II File Edit View Node Assign Utilities Options Window Help
Ref: 150.0ns
Time: 130.8ns
Interval: -19.2ns
150.0ns
60.0ns
120.0ns
180.0ns
240.0ns
300.0ns
360.0ns
420.0ns
Name:
Value:

```
out   0
D     -       0000111100001111
S     B 0101  0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1100 1101 1110 1111
```