Week 4

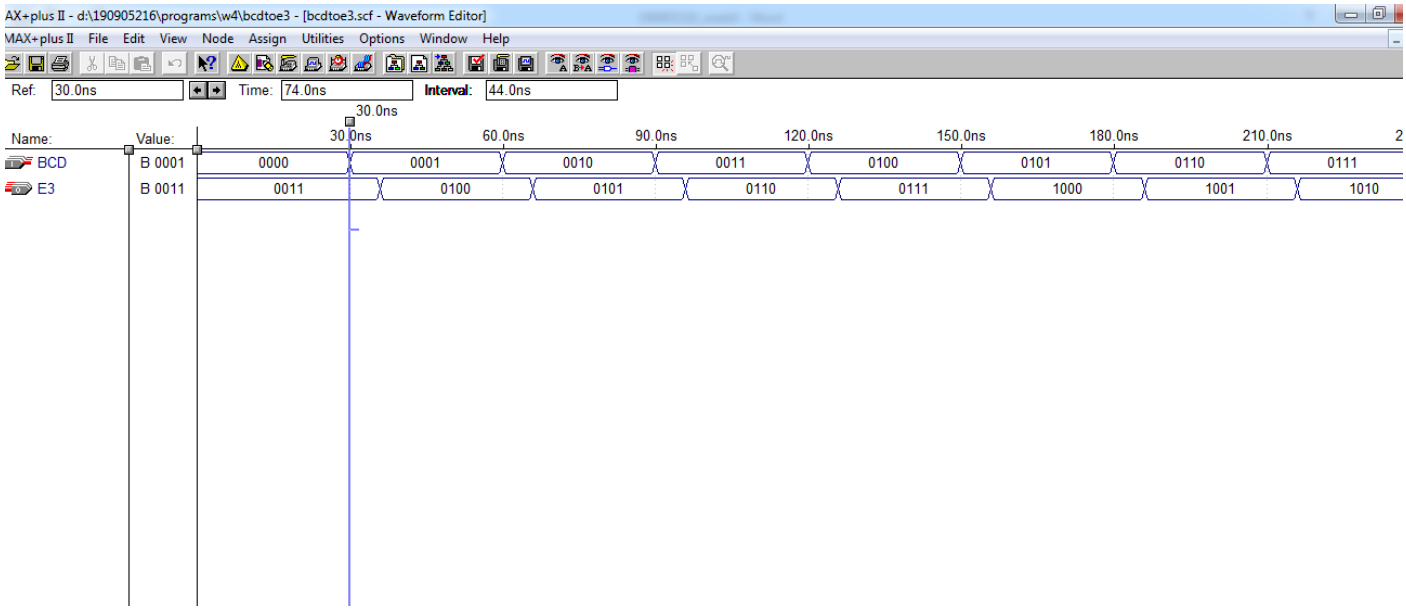
Q1: BCD to Excess 3 Using Mux

```
module bcd2e3(E3, BCD);
input [3:0] BCD;
wire [3:0] BCD;
output [3:0] E3;
reg [3:0] E3;
wire t3, t2, t1, t0;
m8to1 m1(t3, {1'b0, 1'b0, 1'b0, 1'b1, 1'b1, BCD[0], 1'b0, 1'b0}, BCD[3:1]);
m8to1 m2(t2, {1'b0, 1'b0, 1'b0, BCD[0], 1'b0, ~BCD[0], 1'b1, BCD[0]}, BCD[3:1]);
m8to1 m3(t1, {1'b0, 1'b0, 1'b0, ~BCD[0], BCD[0], ~BCD[0], BCD[0], ~BCD[0]}, BCD[3:1]);
assign t0 = ~BCD[0];
always@(BCD)
begin
E3[0] = t0;
E3[1] = t1;
E3[2] = t2;
E3[3] = t3;
end
endmodule

module m8to1(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
```

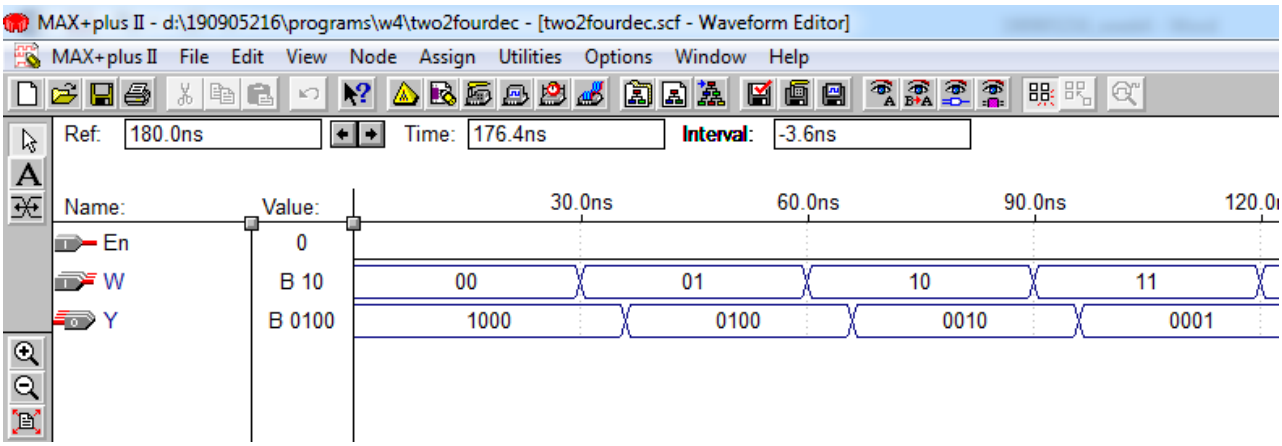
Waveform for BCD to Excess-3



Q2:

```
module two2fourdec(W,En,Y);
input[1:0]W;
input En;
output [0:3]Y;
reg [3:0]Y;
always@(W or En)
begin
if(En==0) //Active Low Enable Input
case(W)
0: Y=4'b1000;
1: Y=4'b0100;
2: Y=4'b0010;
3: Y=4'b0001;
endcase
else
Y=4'b0000;
end
endmodule
```

Waveform for 2 to 4 Decoder (Active Low Enable Input)



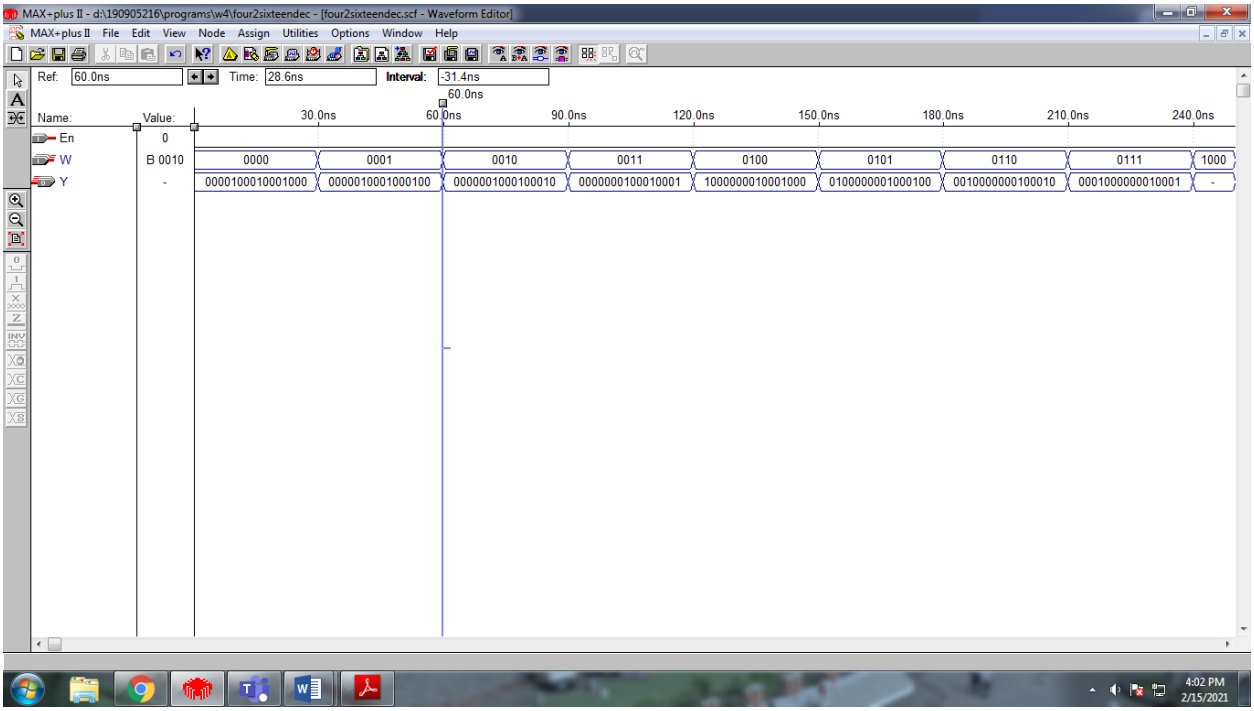
4 to 16 Decoder:

```
module four2sixteendec(W,En,Y);
input [3:0]W;
input En;
output [0:15]Y;
wire[0:3]M;
dec2to4 dec1(W[3:2],En,M[0:3]);
dec2to4 dec2(W[1:0],M[0],Y[0:3]);
dec2to4 dec3(W[1:0],M[1],Y[4:7]);
dec2to4 dec4(W[1:0],M[2],Y[8:11]);
dec2to4 dec5(W[1:0],M[3],Y[12:15]);
endmodule

module dec2to4(W,En,Y);
input[1:0]W;
input En;
output [0:3]Y;
reg [0:3]Y;
always@(W or En)
begin
if(En==0)
case(W)
0: Y=4'b1000;
1: Y=4'b0100;
2: Y=4'b0010;
3: Y=4'b0001;
endcase
else
Y=4'b0000;
```

end

endmodule



PTO->

Q3: 16 to 4 Priority Encoder

```
module penc16to4(w,y,z);

input [15:0]w;

output [3:0]y;

reg [3:0] y;

output z;

reg z;

integer k;

always @(w)

begin

z = 0;

if(w == 0)

y = 0;

else

begin

for(k = 0 ; k < 16 ; k = k + 1)

begin if(w[k] == 1)

y = k;

end

z = 1;

end

end

endmodule
```

---**Waveform**--