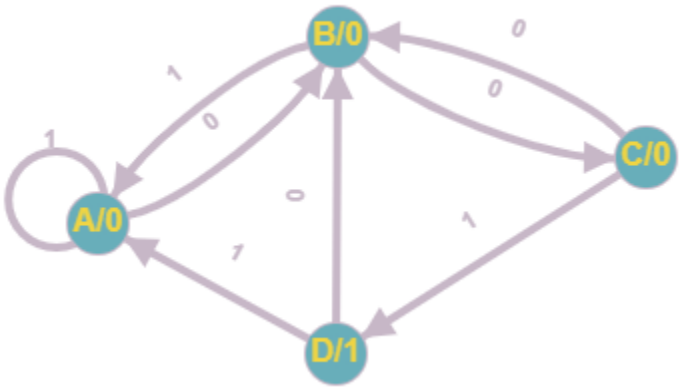
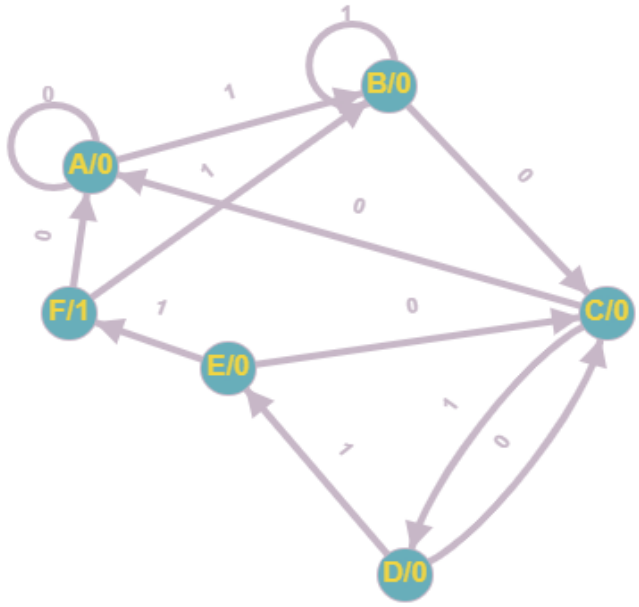
Brennen Green
Prelab 5
CPE282

0 0 1



1 0 1 1 1



Brennen Green
Prelab 5
CPE282
Verilog

a. 0 0 1

```
module FSM001(input clk, input Resetn, input w, output out);
    reg[2:1]y, Y;
    parameter [2:1] A = 2'b00, B=2'b01, C=2'b10, D=2'b11;

    always @(w,y)
        case(y)
            A: if(w==0) Y=B;
                else Y=A;
            B: if(w==0) Y=C;
                else Y=A;
            C: if(w==1) Y=D;
                else Y=B;
            D: if(w==1) Y=A;
                else Y=B;
        endcase

    always @(posedge Clock)
        if(Resetn == 1) y <= A;
        else y <= Y;

    assign out = (y==D);
endmodule
```

b. 1 0 1 1 1

```
module FSM10111(input clk, input Resetn, input w, output out);
    reg[3:1]y, Y;
    parameter [3:1] A = 3'b000, B=3'b001, C=3'b010, D=3'b011, E=3'b100, F=3'b101;

    always @(w,y)
        case(y)
            A: if(w==0) Y=A;
                else Y=B;
            B: if(w==0) Y=C;
                else Y=B;
            C: if(w==0) Y=A;
                else Y=D;
            D: if(w==0) Y=C;
                else Y=E;
            E: if(w==0) Y=C;
                else Y=F;
            F: if(w==0) Y=A;
                else Y=B;
        endcase

    always @(posedge Clock)
        if(Resetn == 1) y <= A;
        else y <= Y;

    assign out = (y==F);
endmodule
```

Brennen Green
Prelab 5
CPE282

16 bit shift register

```
module shift16(R, L, w, Clock, Q);
    input [15:0] R;
    input L, w, Clock;
    output reg [15:0] Q;

    always(@posedge Clock)
        if(L)
            Q <= R;
        else
        begin
            Q[0] <= Q[1];
            Q[1] <= Q[2];
            Q[2] <= Q[3];
            Q[3] <= Q[4];
            Q[4] <= Q[5];
            Q[5] <= Q[6];
            Q[6] <= Q[7];
            Q[7] <= Q[8];
            Q[8] <= Q[9];
            Q[9] <= Q[10];
            Q[10] <= Q[11];
            Q[11] <= Q[12];
            Q[12] <= Q[13];
            Q[13] <= Q[14];
            Q[14] <= Q[15];
            Q[15] <= w;
        end
endmodule
```

4 bit counter

```
module count4(input Reset, input Clock, input en, output reg [3:0] Q);
    always @(posedge Clock)
        begin
            if (Reset == 1)
                Q<=0;
            else if (en)
                Q <= Q + 1;
        end
endmodule
```