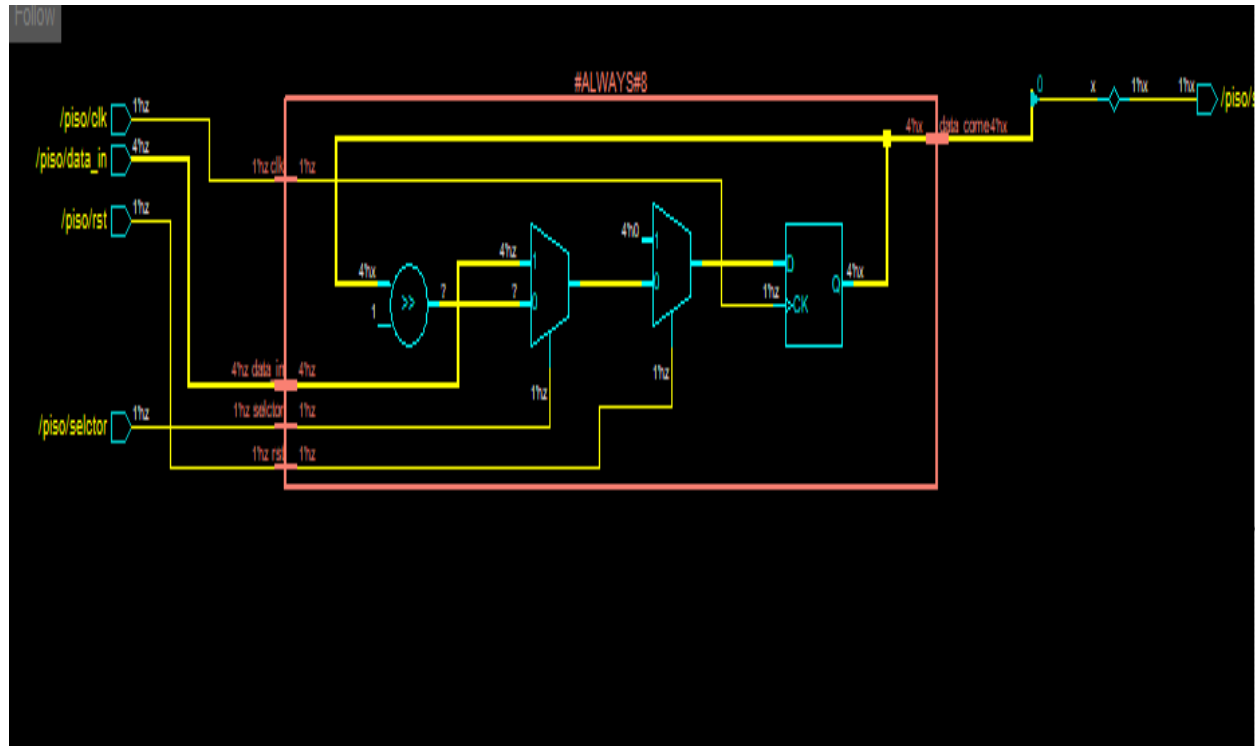
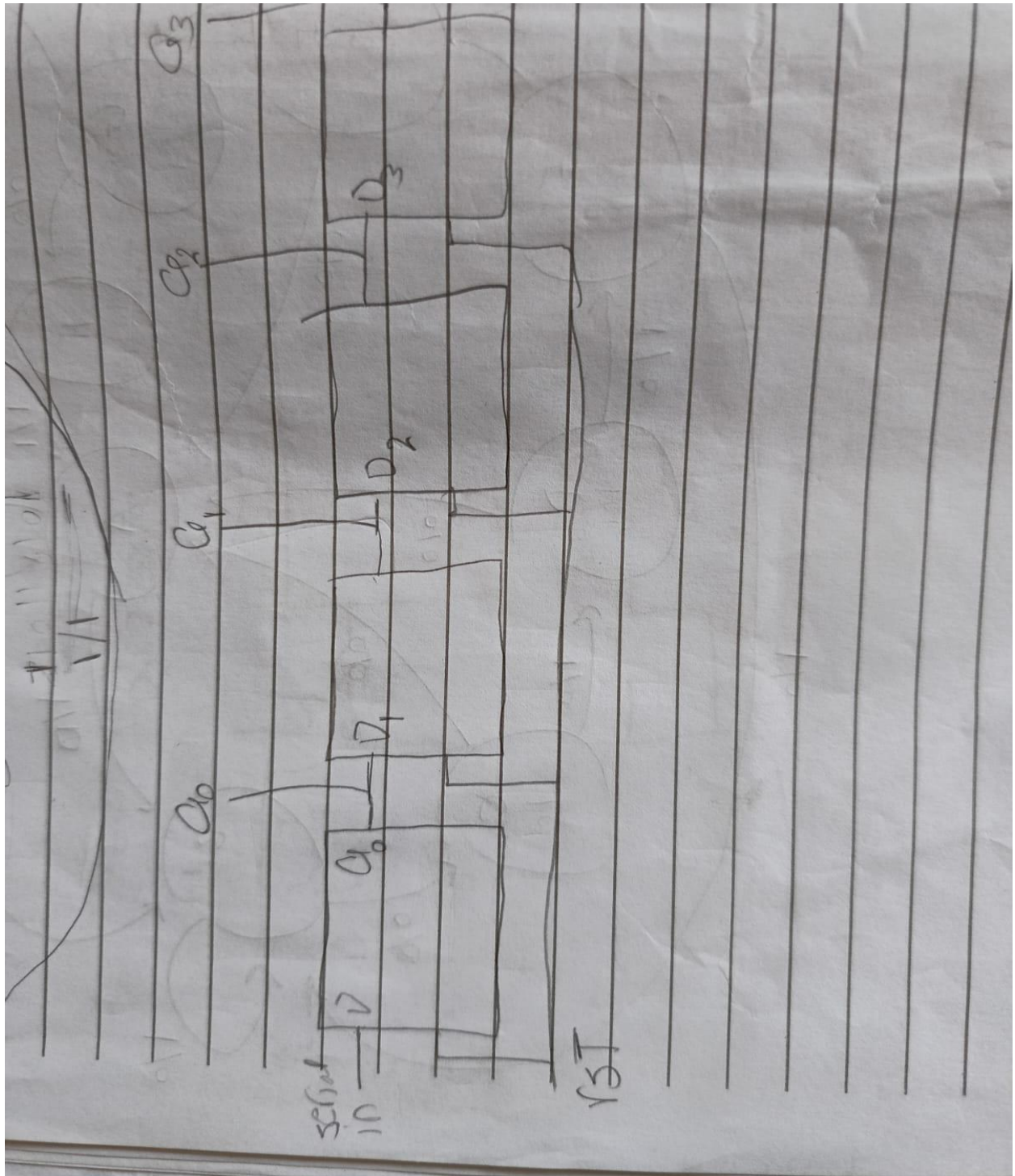


## Continuo on lap2

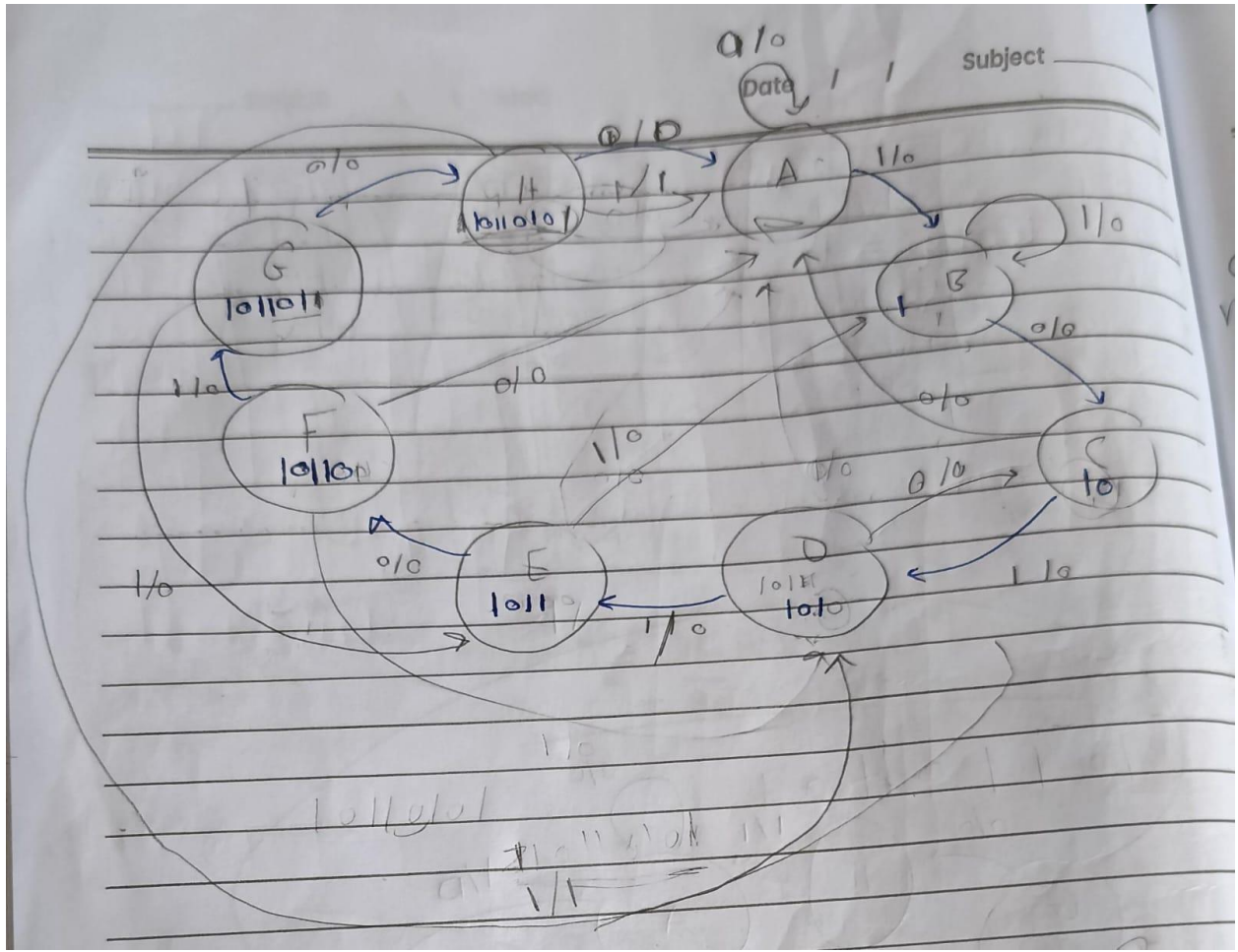
### PISO Schematic



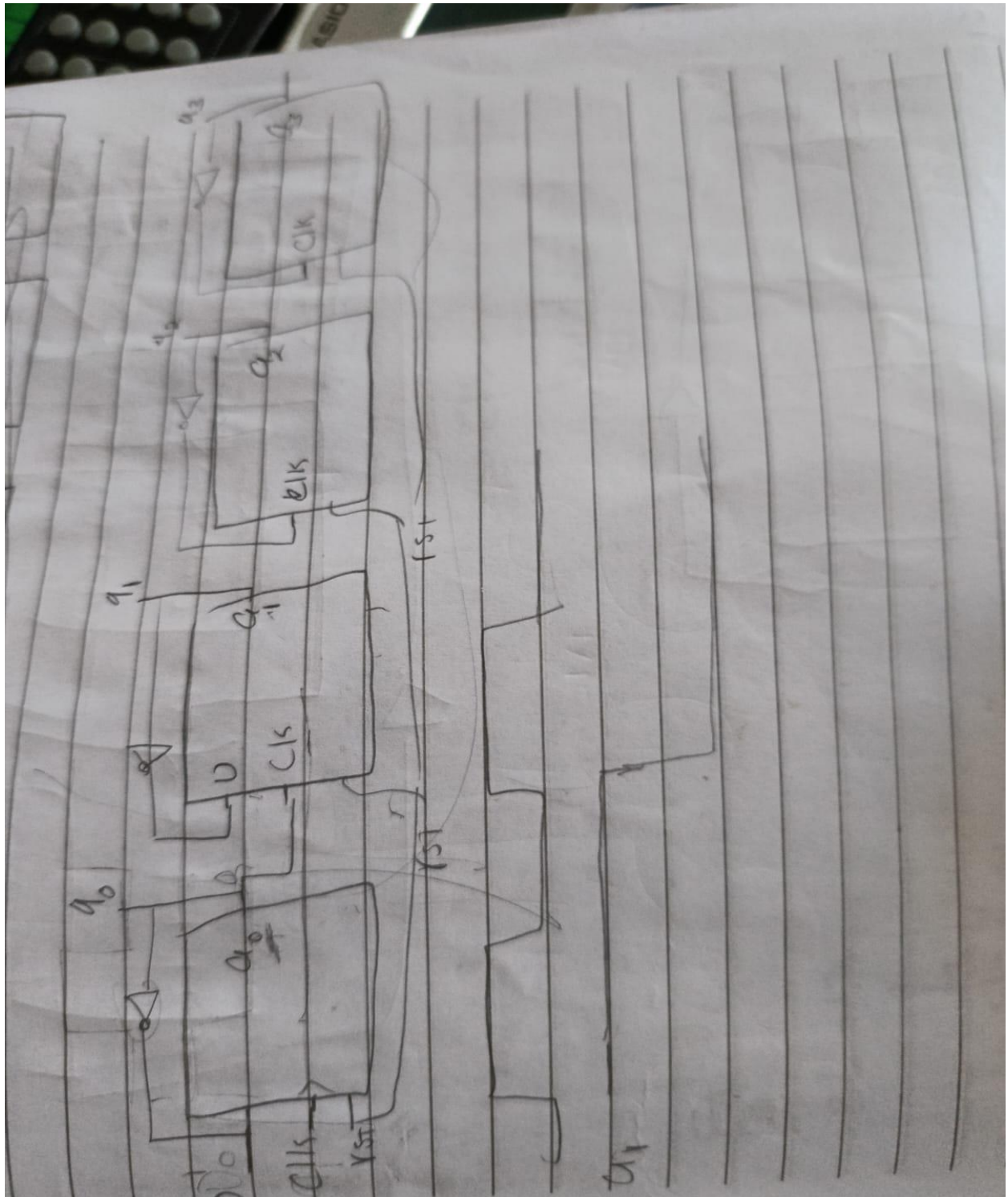
## SIPO schem



## Fsm2



Divide by 2,4,8,16



## Fsm2\_style

```
input reset,go,clk,
output reg op
);

localparam A=3'b000;
localparam B=3'b001;
localparam C=3'b010;
localparam D=3'b011;
localparam E=3'b100;
localparam F=3'b101;
localparam G=3'b110;
localparam H=3'b111;

reg [2:0] next_state,present_state;

always@(posedge clk)
begin
if(reset)
present_state<=0;
else
```

```

21 else
22 present_state<=next_state;
23 end
24
25 always@*
26 begin
27 case(present_state)
28 A:
29 begin
30 if(go)
31 next_state=B;
32 else
33 next_state=A;
34 end
35 B:
36 begin
37 if(go)
38 next_state=B;
39 else
```

```
40         next_state=C;
41     end
42 C:
43     begin
44         if(go)
45             next_state=D;
46         else
47             next_state=A;
48         end
49 D:
50     begin
51         if(go)
52             next_state=E;
53         else
54             next_state=C;
55         end
56 E:
57     begin
58         if(go)
59             next_state=B;
```

---

```
59         next_state=B;
60     else
61         next_state=F;
62     end
63 F:
64     begin
65         if(go)
66             next_state=G;
67         else
68             next_state=A;
69         end
70 G:
71     begin
72         if(go)
73             next_state=E;
74         else
75             next_state=H;
76         end
77 H:
78     begin
```

---

```
79         if(go)
80             next_state=D;
81         else
82             next_state=A;
83         end
84     default:
85     begin
86         next_state=present_state;
87         present_state=A;
88     end
89 endcase
90 end
91
92 always@*
93 begin
94
95     case (present_state)
96     A:op=0;
97     B:op=0;
98     C:op=0;
```

```
99     D:op=0;
100     E:op=0;
101     F:op=0;
102     G:op=0;
103     H:op=1;
104     endcase
105
106 end
107
108 endmodule
```

## Edit for fifo code

```
2  input clk,rst,en_w,en_r,
3  input [31:0]data_in,
4  output reg full_flag,empty_flag,
5  output reg [7:0]data_out
6  );
7
8  reg [31:0] mem[0:7];
9  integer i;
10
11  reg [2:0] write_pointer,read_pointer,count;
12
13  always@(posedge clk)
14  begin
15  if(rst)
16  begin
17  for(i=0;i<8;i=i+1)
18  begin
19  mem[i]<='b0;
20  end
21  empty_flag<=1;
```

```
21  empty_flag<=1;
22  full_flag<=0;
23  write_pointer<=0;
24  read_pointer<=0;
25  data_out<=0;
26  count<=0;
27  end
28
29  else if(en_w && !full_flag && count<8) //logically *****
30  begin
31  mem[write_pointer]<=data_in;
32  write_pointer<=write_pointer+1;
33  if(count==7)
34  begin
35  empty_flag<=0;
36  full_flag<=1;
37  end
38  else
```



```
40 begin
41     count=count+1;
42     empty_flag<=0;
43     full_flag<=0;
44 end
45
46 end
47
48 else if(en_r && !empty_flag && count>=0)
49 begin
50     data_out<=mem[read_pointer];
51     mem[read_pointer]<='b0;
52     read_pointer<=read_pointer+1;
53     if(count==3'b000)
54     begin
55         empty_flag<=1;
56         full_flag<=0;
57     end
58
```

---

```
59     else
60
61     begin
62         count=count-1;
63         empty_flag<=0;
64         full_flag<=0;
65     end
66
67 end
68 end
69
70 endmodule
```

---