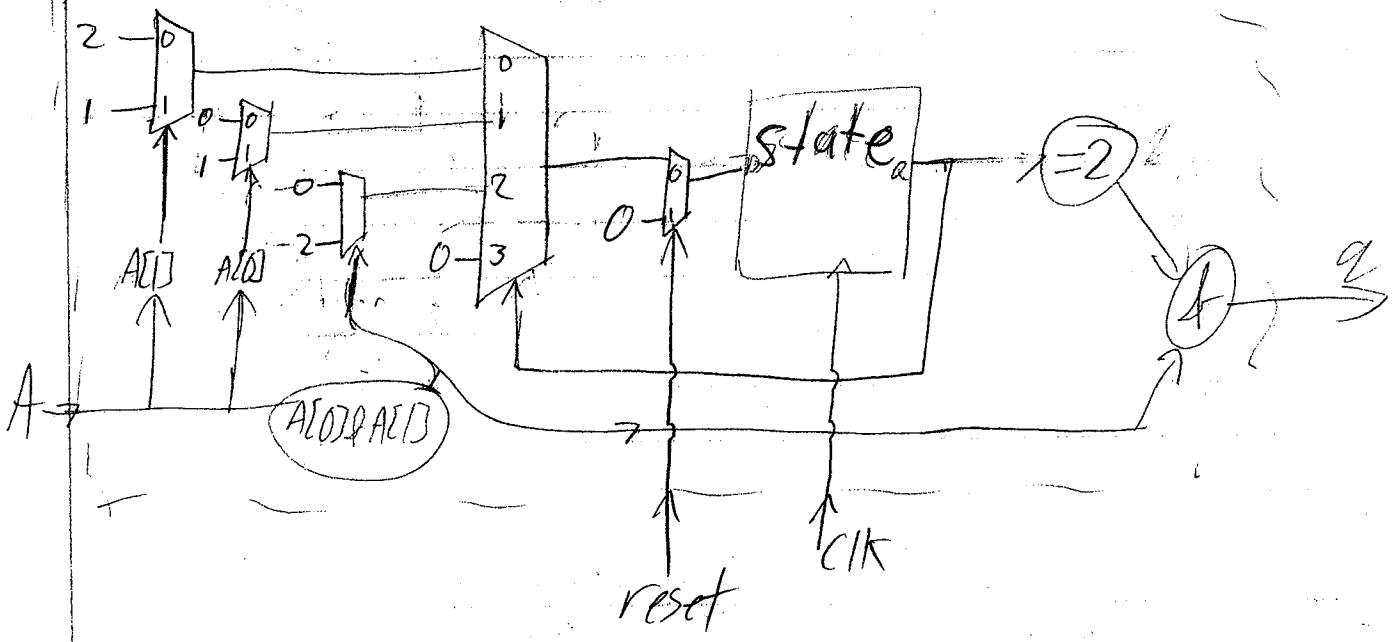
CS150 HW2 solutions Fall 2013

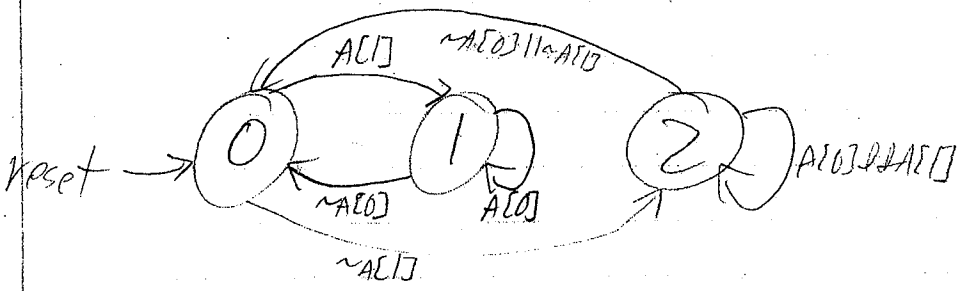
S. Twigg

Problem 1.

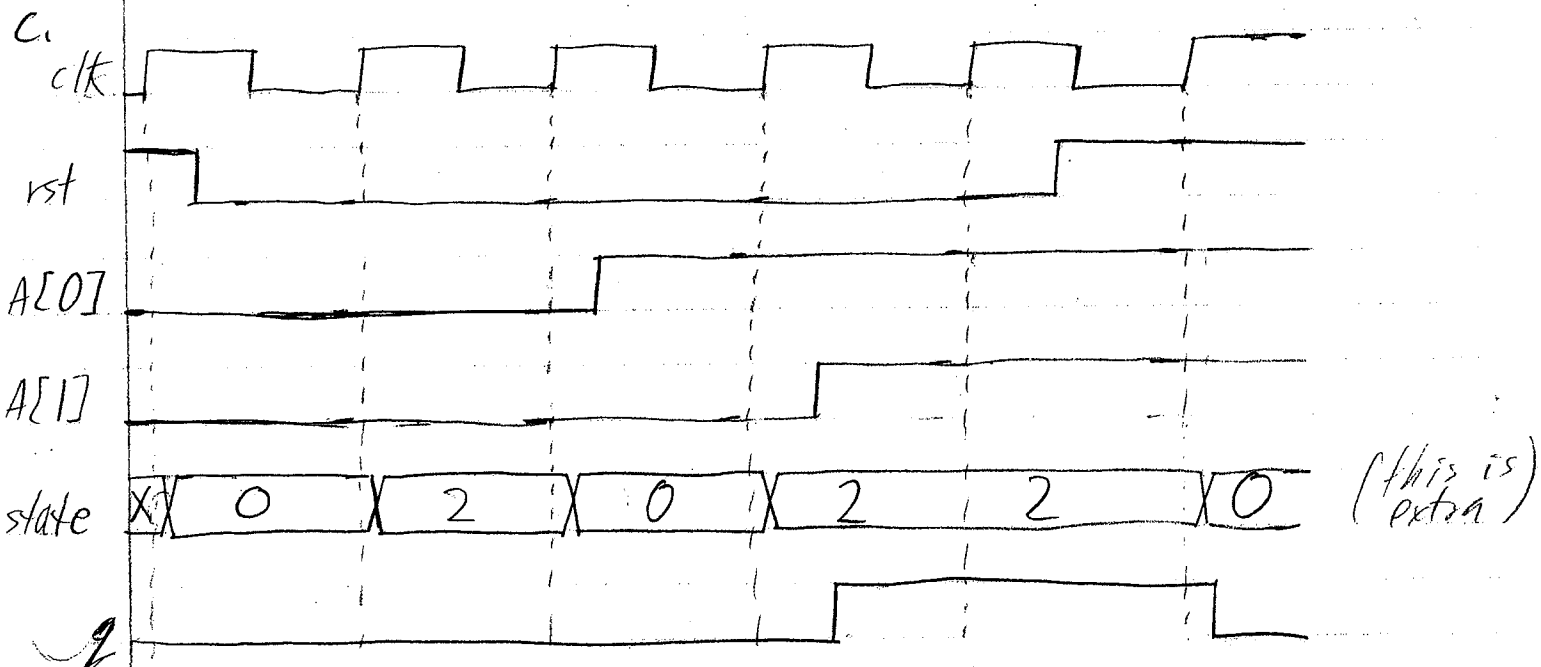
a.



b.

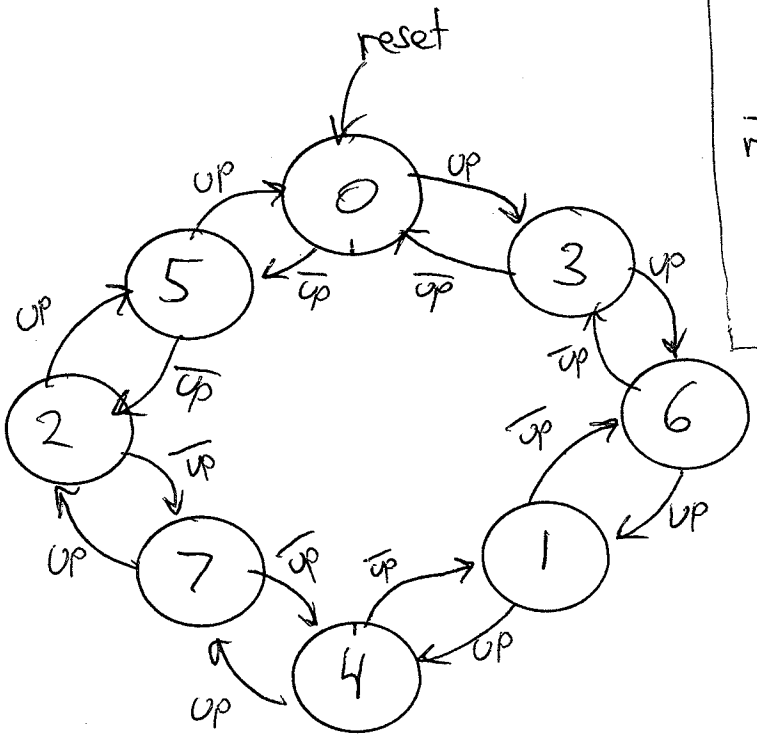


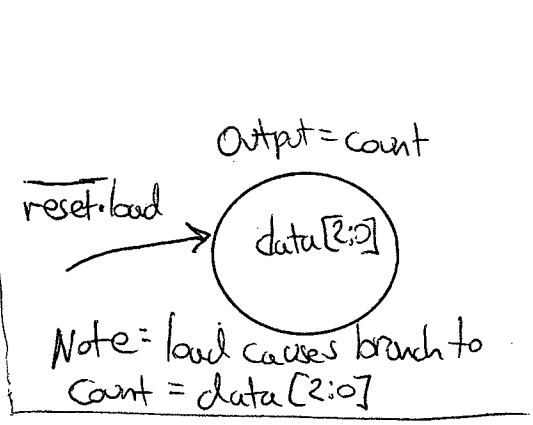
c.



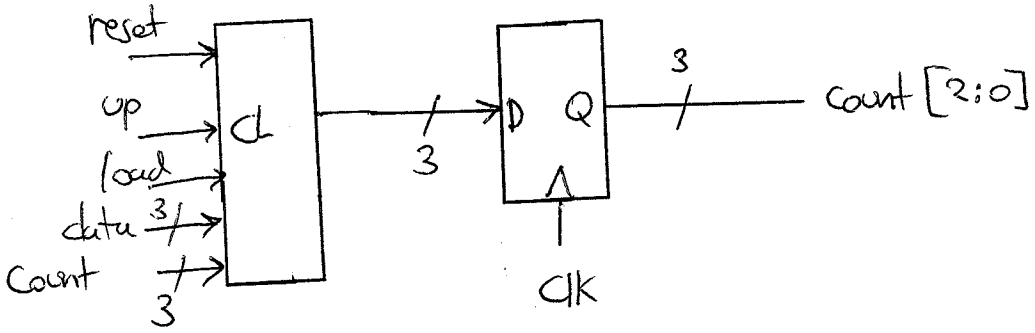
Problem 2

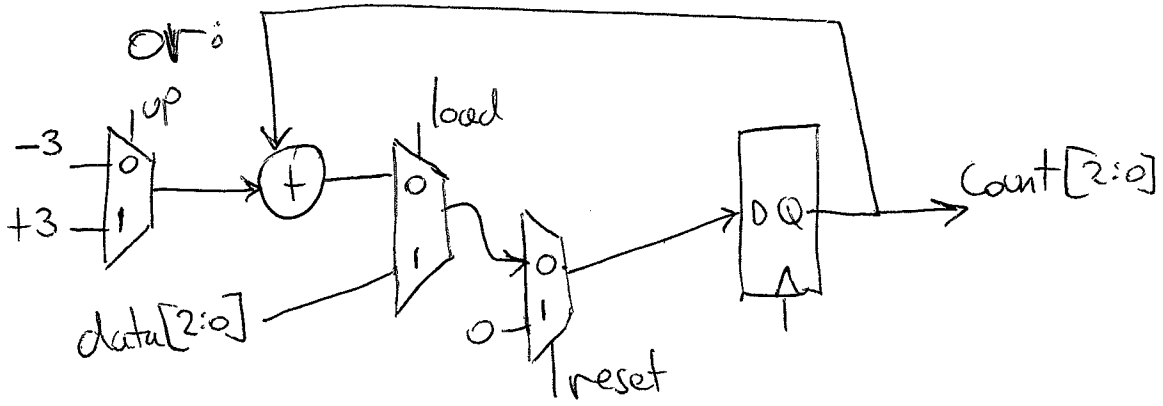
a)


reset
0
1
2
3
4
5
6
7
up
$\overline{up}$


Output = count
$\overline{reset \cdot load}$
data[2:0]
Note: load causes branch to
count = data[2:0]

b)


reset
up
load
data
3
Count
3
CL
3
D
Q
3
count [2:0]
clk


or
up
-3
+3
0
1
+
load
0
1
data[2:0]
0
1
0
1
reset
D
Q
count[2:0]

Problem 2c

version1

```
module hw2_2FSM_v1(
        input clk, input reset,
        input up, input load, input [2:0] data,
        output [2:0] count
    );
    reg [2:0] state;

    always @(posedge clk)
    begin
        if(reset)
            state <= 0;
        else if(load)
            state <= data;
        else if(up)
            state <= state + 3;
        else
            state <= state - 3;
    end

    assign count = state;

endmodule
```

---

version2

```
module hw2_2FSM_v2(
        input clk, input reset,
        input up, input load, input [2:0] data,
        output [2:0] count
    );
    reg [2:0] state, nextstate;

    always @(posedge clk)
    begin
        if(reset) state <= 0;
        else state <= nextstate
    end

    always @(*)
    begin
        nextstate <= state
        if(load)
            nextstate <= data;
        else if(up)
            nextstate <= state + 3;
        else
            nextstate <= state - 3;
    end

    assign count = state;

endmodule
```

} State update

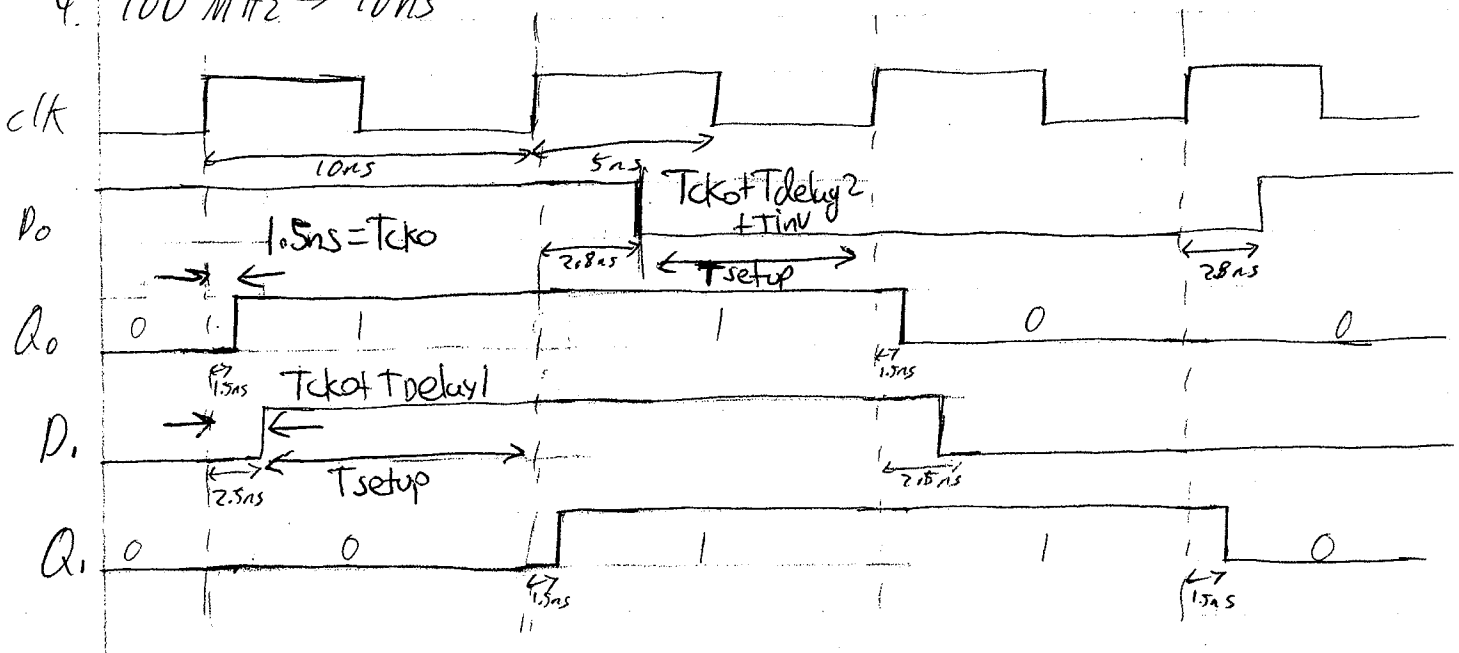
} Next State Decode

Problem 3

```
module summer(
        input clk, input rst,
        output [7:0] total, output done
    );
    reg [3:0] addr_counter;
    reg [7:0] accumulator;
    wire [3:0] mem_out;

    always @(posedge clk)
    begin
        if(rst) addr_counter <= 0;
        else addr_counter <= addr_counter + 1;
    end

    RAM16x4S #(
        // These constants can be anything but this part should be here.
        .INIT_00(16'hffff), // INIT for bit 0 of RAM
        .INIT_01(16'h0000), // INIT for bit 1 of RAM
        .INIT_02(16'h0000), // INIT for bit 2 of RAM
        .INIT_03(16'h0000), // INIT for bit 3 of RAM
    ) RAM16x4S_inst (
        .O0(mem_out[0]),
        .O1(mem_out[1]),
        .O2(mem_out[2]),
        .O3(mem_out[3]),
        .A0(addr_counter[0]),
        .A1(addr_counter[1]),
        .A2(addr_counter[2]),
        .A3(addr_counter[3]),
        .D0(0), // can be whatever
        .D1(0), // can be whatever
        .D2(0), // can be whatever
        .D3(0), // can be whatever
        .WCLK(0), // can be whatever
        .WE(0) // THIS MUST BE HERE
    );

    always @(posedge clk)
    begin
        if(rst) accumulator <= 0;
        else accumulator <= accumulator + mem_out;
    end

    assign total = accumulator;
    assign done = addr_counter == 4'b111;

endmodule
```

optional

4. 100 MHz → 10ns



b, 2.8ns + 1.0ns = 3.8ns minimum clock period

b. Min period = $\max\{T_{cko}+T_{delay1}+T_{setup},\ T_{cko}+T_{delay2}+T_{inv}+T_{setup}\}$

$= \max\{3.5,\ 3.8\text{ns}\}$