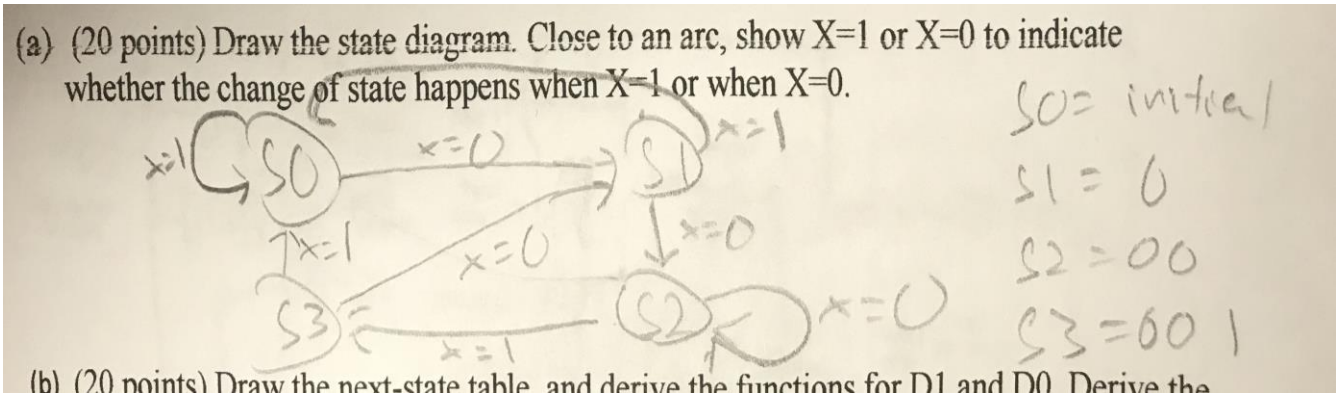
Assignment #6 – Digital Logic Design II – Sequential Logic

CDA 3100, Computer Organization I

Due: 12/8/18, please submit an electronic version on Canvas. TROY ALLEN

Problem 1 (50 points) Design a circuit that has two inputs, clk and X, and produces one output O. X may change every clock cycle, and the change happens at the falling edge. The circuit samples the input at every rising edge of the clock. If the input is 1, consider as read a 1, else read a 0. O is 1 (for one clock cycle, from positive edge to positive edge) if the last three bits read are 001, where 1 is the most recent bit.



(b) (20 points) Draw the next-state table, and derive the functions for D1 and D0. Derive the output function.

S0 = 00, S1 = 01, S2 = 10, S3 = 11

D1 = (Q1&~Q0) | (~Q1&Q0&~X) D0 = (~Q1&~Q0&~X) | (Q1&~Q0&X) | (Q1&Q0&~X) O = Q1&Q2

Q1	Q0	X	D1	D0
0	0	0	0	1
0	0	1	0	0
0	1	0	1	0
0	1	1	0	0
1	0	0	1	0
1	0	1	1	1

1	1	0	0	1
1	1	1	0	0

(c) (10 points) Complete the Verilog module for this circuit. Use the following code as a template.

```
module HW6P2 (clk, X, O);
        input clk, X;
        output O;

        wire D1, D0, Q1, Q0, Q1bar, Q0bar;

        assign D0 = (~Q1&~Q0&~X) | (Q1&~Q0&X) | (Q1&Q0&~X);
        Dff1 C0 (D0, clk, Q0, Q0bar);

        assign D1 = (Q1&~Q0) | (~Q1&Q0&~X);

        Dff1 C1 (D1, clk, Q1, Q1bar);

        assign O = Q1&Q2;

endmodule
```

Problem 2 (50 points) Design a MIPS processor supporting **only** the **`R-type`** and the **`AAA rs, rt, a`** instruction. The `AAA rs, rt, a` is an **`I-type`** instruction that does the following:

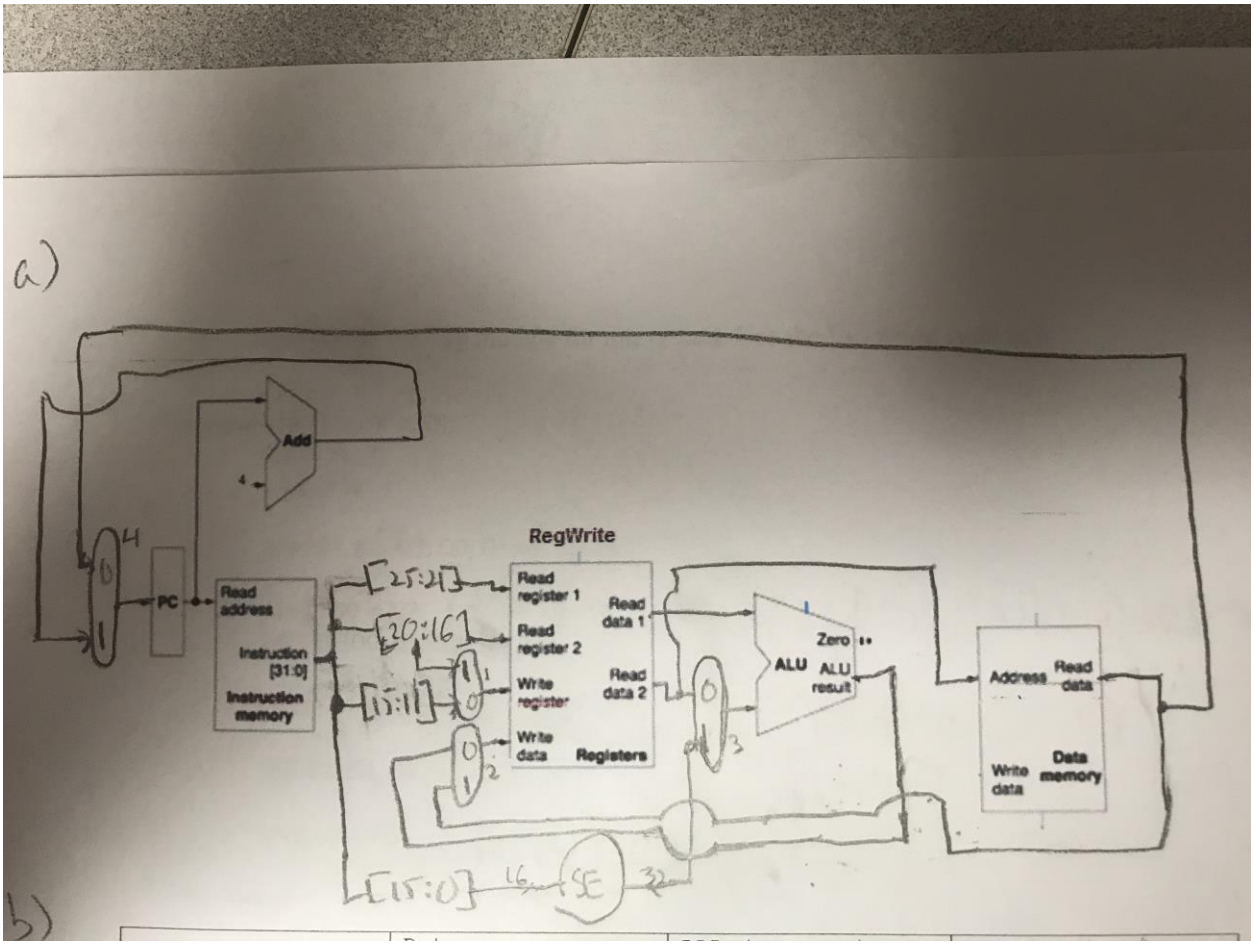
> **If `rs` equal to a (after sign extension), the next `PC` will be the data memory value at address `rt`; otherwise, `rt` will be the data memory value at address `rt`.**

For this problem, assume the `opcode` of `R-type` is 000000 and the `opcode` of `AAA` is 100000.

(a) (30 points) Please design the datapath of this processor, add 2-1 MUX when necessary. Please show clearly the indices of the bits near the wires**.** Please give clear index to the 2-1 MUX starting from 1.

(b) (20 points) Please determine the values of `RegWrite` and the control signals of the 2-1 MUX by filling in the table. In case of "don't care", write down "X." Assume other control signals have been generated correctly.

(c) (Extra 10 points) Please write down the logic functions of `RegWrite` and the control signals of the 2-1 MUX. Certain bits in the instruction can be denoted, for example, as `instuct[31]`. The ALU zero output can be denoted simply as "zero." **When deriving the logic functions, if the value of a signal is "don't care" under a certain condition, assume it should be 0.**



	R-type	AAA (rs == a)	AAA (rs != a)
RegWrite	1	0	1
MUXCtrl1	0	X	1
MUXCtrl2	0	X	1
MUXCtrl3	0	1	1
MUXCtrl4	1	0	1

c)

RegWrite = ~Instruct[31] | zero

MUXCtrl1 = Instruct[31] & zero

MUXCtrl2 = Instruct[31] & zero

MUXCtrl3 = Instruct[31] | zero

MUXCtrl4 = ~Instruct[31] & zero