Exam - Computer Architecture Unit I [08/02/2023] (B)

Surname: ______________________________ Name: __________________________

Student ID Number (Matricola): ____

DSA Students should solve only the first 4 exercises (grade will be scaled accordingly)

Exercise 1 (8 points)

Design a sequential circuit with two inputs x1, x0, that encode the characters O, R, C as follows:

x1, x0	carattere
00	O
01	R
1-	C

The circuits has 2 outputs z1 and z0. The circuit outputs z1=1 when it receives on input the sequence ORCO, and outputs z0=1 when it receives on input the sequence ORO. Overlaps are allowed. Draw the circuit.

Solution:

Same state transition diagram of text A. Just replace A->O, L->R, F->C and the states accordingly (R state is here called ‘S’). Invert z1 and z0.

State encoding:

S	**00**
O	**01**
OR	**10**
ORC	**11**

Output and next state table:

CS	S_1	S_0	x_1	x_0	NS	S_1'	S_0'	z1	z0
S	0	0	0	0	O	0	1	0	0
S	0	0	0	1	S	0	0	0	0
S	0	0	1	-	S	0	0	0	0
O	0	1	0	0	O	0	1	0	0
O	0	1	0	1	OR	1	0	0	0
O	0	1	1	-	S	0	0	0	0
OR	1	0	0	0	O	0	1	0	**1**
OR	1	0	0	1	S	0	0	0	0
OR	1	0	1	-	ORC	1	1	0	0
ORC	1	1	0	0	O	0	1	**1**	0
ORC	1	1	0	1	S	0	0	0	0
ORC	1	1	1	-	S	0	0	0	0

Equations and circuit the same as text A (just invert z1 and z0).

Exercise 2 (6 points) A combinational circuit has a 4 bit input A= $a_3a_2a_1a_0$ representing a two’s complement number, and outptus Z=z_1z_0 as an unsigned binary number, such that:

Z=0 se $-7 \leq A < -3$

Z=1 se 4≤A≤7
Z=2 se -3≤A<4
Z=3 se A=-8

Write down:

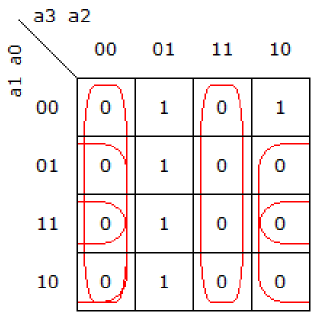
- The corresponding truth table
- The minimal POS form for z_0
- The all-NAND and all-NOR form for z_0 (NOT gates are allowed, i.e., you don’t need to translate NOT with NAND/NOR)
- z_0 using only three 2:1 multiplexers

Solution:

Truth table:

a_3	a_2	a_1	a_0	z1	z0
0	0	0	0	1	0
0	0	0	1	1	0
0	0	1	0	1	0
0	0	1	1	1	0
0	1	0	0	0	1
0	1	0	1	0	1
0	1	1	0	0	1
0	1	1	1	0	1
1	0	0	0	1	1
1	0	0	1	0	0
1	0	1	0	0	0
1	0	1	1	0	0
1	1	0	0	0	0
1	1	0	1	1	0
1	1	1	0	1	0
1	1	1	1	1	0

Minimal POS form for z_0:



a1 a0 \ a3 a2	00	01	11	10
00	0	1	0	1
01	0	1	0	0
11	0	1	0	0
10	0	1	0	0

z_0= (a3+a2)*(~a3+~a2)*(a2+~a0)*(a2+~a1)

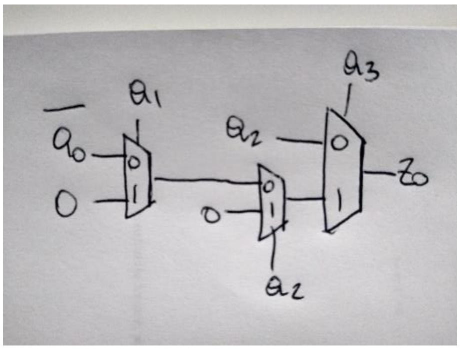
all-NAND form for z_0:

z_0= ~(~(~(~a3*~a2)*~(a3*a2)*~(~a2*a0)*~(~a2*a1)))

all-NOR form for z_0:

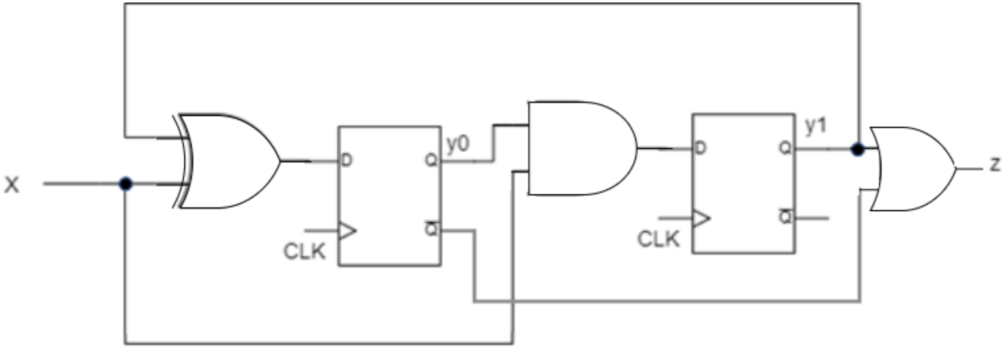
z_0= ~(~(a3+a2)+~(~a3+~a2)+~(a2+~a0)+~(a2+~a1))

z_0 using MUX:



Exercise 3 (5 points)

Analyze the state machine shown in the figure. Write down next state and output tables and draw the state transition diagram.



Solution:

It is a Moore FSM (output does not depend on the input).

Outputs table + state encoding

z=y0'+y1

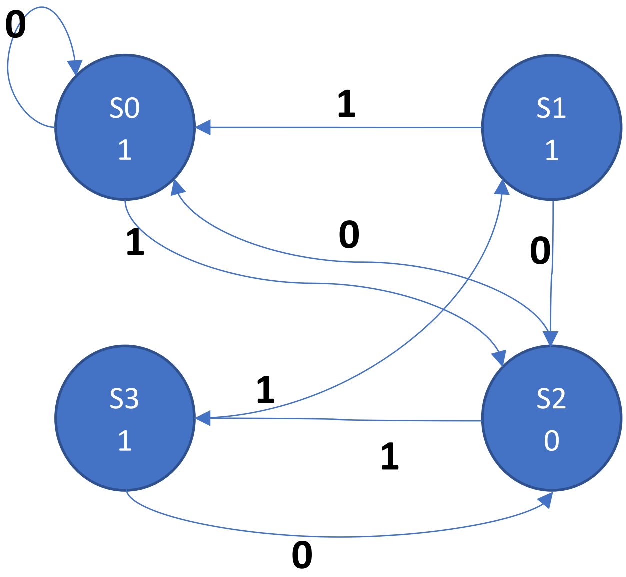
y0	y1	z	Stato
0	0	1	S0
0	1	1	S1
1	0	0	S2
1	1	1	S3

Next state table

y0' = x XOR y1

y1' = y0 x

y0	y1	CS	x	y0'	y1'	NS
0	0	S0	0	0	0	S0
0	0	S0	1	1	0	S2
0	1	S1	0	1	0	S2
0	1	S1	1	0	0	S0
1	0	S2	0	0	0	S0
1	0	S2	1	1	1	S3
1	1	S3	0	1	0	S2
1	1	S3	1	0	1	S1



Exercise 4 (3 points)

Draw the state transition diagram described by the following SystemVerilog code:

```
module fsm2(input logic clk, reset,
            input logic a, b,
            output logic y);
   logic [1:0] state, nextstate;
   parameter S0 = 2'b00;
   parameter S1 = 2'b01;
   parameter S2 = 2'b10;
   parameter S3 = 2'b11;
   always_ff @(posedge clk, posedge reset)
     if (reset) state <= S0;
     else state <= nextstate;
   always_comb
     case (state)
       S0: if (a | b) nextstate = S2;
           else nextstate = SO;
       S1: if (a | b) nextstate = S2;
           else nextstate = SO;
       S2: if (a & b) nextstate = S3;
           else nextstate = SO;
       S3: if (a ^ b) nextstate = S3;
           else nextstate = S1;
     endcase
   assign y = (state== S1) | (state== S3);
endmodule
```

Solution:

Similar to track A.

Exercise 5 (4 points)

Given X = 775 and Y = -1524 represented in base 10:

- Convert them to two's complement numbers using the minimum number of bits necessary to represent both values
- Compute X+Y and Y-X
- Convert the results to base 10 number and check their correctness
- Convert X and Y to IEEE 754 half-precision representation

Solution:

Base 2 conversion:

775   = 001100000111
1524 = 010111110100 => -1524 = 101000001100

Sum and difference in base 10:

```
X+Y = 101000001100+
        001100000111=
------------------------------
        110100010011 = -749 = 775-1524
```

-775 = 110011111001

```
Y-X = 101000001100+
       110011111001=
------------------------------
       011100000101 = Overflow! -1524 – 775 = -2299 (11 bit are not enough, they can only represent up to -2048)
```

Half-precision conversion

775 = 001100000111.0 = 1.100000111 * 2^9

X = 775:

- Sign = 0
- Exponent = 9_{10}
- Exponent + bias = $9_{10}+15_{10} = 24_{10} = 11000_2$
- Fraction = 100000111
- IEEE: 0 11000 1000001110

1524 = 010111110100.0 = 1.0111110100 * 2^{10}

Y = -1524

- Sign = 1
- Exponent = 10
- Exponent + bias = 10+15 = 25 = 11001
- Fraction = 0111110100
- IEEE = 1 11001 0111110100

Exercise 6 (4 points)

Given the expression $f = \left(\overline{c\left(\overline{abd} + c\right)} + \bar{e}\right) \oplus (e + cd)$ simplify it and bring to minimal POS form.

Solution:

$$f = \left(\overline{c\left(\overline{abd} + c\right)} + \bar{e}\right) \oplus (e + cd)$$

$$= \left(\overline{\mathrm{c}\overline{\mathrm{abd}} + \mathrm{c}} + \bar{e}\right) \oplus (e + cd)$$

$$= (\bar{c} + \bar{e}) \oplus (e + cd)$$

$$= \overline{(\bar{c} + \bar{e})}(e + cd) + (\bar{c} + \bar{e})\overline{(e + cd)} =$$

$$= ce(e + cd) + (\bar{c} + \bar{e})\bar{e} \cdot \overline{cd} = ce + dce + \bar{c}\bar{e}\overline{cd} + \bar{e}\overline{cd} = ce + \bar{e}\bar{c} + \bar{e}\bar{d} =$$

$$= (c + \bar{e})(c + \bar{c})(e + \bar{e})(e + \bar{c}) + \bar{e}\bar{d} = (c + \bar{e})\left(c + \bar{e} + \bar{d}\right)(e + \bar{c} + \bar{e})\left(e + \bar{c} + \bar{d}\right)$$

$$= (c + \bar{e})\left(c + \bar{e} + \bar{d}\right)\left(e + \bar{c} + \bar{d}\right) = (c + \bar{e})\left(e + \bar{c} + \bar{d}\right)$$