**Homework Problem Set #5**

**Note:**

**Submit each circuit file separately, give a meaningful name to each file! Do not Zip!**

**Q1. (6 points) A full-adder is a combinational circuit that forms the arithmetic sum of three input bits. It consists of three inputs, x, y, z, and two outputs, C and S. Two of the input, that is, x and y, represent the two significant bits to be added. The third input, z, represents the carry from the previous lower significant position. The output S denotes the sum of two bits and C denotes carry. Answer the following sub-questions.**

1) Construct a truth table for the Full-Adder

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x | y | z | c | s |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

2) Based on the truth table, construct a K-map for the output S and derive a Boolean equation using K-map. Make the equation as simple as possible.

S k-map:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | xy | !xy | !x!y | x!y |
| z | 1 |  | 1 |  |
| !z |  | 1 |  | 1 |

s = xyz + !xy!z + !x!yz + x!y!z

3) Based on the truth table, construct a K-map for the output C and derive a Boolean equation using K-map. Make the equation as simple as possible.

C k-map:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | xy | !xy | !x!y | x!y |
| z | 1 | 1 |  | 1 |
| !z | 1 |  |  |  |

c = xy + yz + xz

4) By algebraic manipulation, show that S can be expressed as the exclusive-OR of the three input variables. That is, show that,

|  |  |
| --- | --- |
| S = x XOR y XOR z | s = xyz + !xy!z + !x!yz + x!y!z |
| S = (x XOR y) XOR z |  |
| S = (x XOR y)!z + !(x XOR y)z |  |
| S = (x!y + !xy)!z + ( !x!y + xy)z |  |
| S = x!y!z + !xy!z + !x!yz + xyz | == xyz + !xy!z + !x!yz + x!y!z |

You can prove by deriving from S = x XOR y XOR z to the answer you got in the question 2)

5) By algebraic manipulation, show that C can be expressed as the following term.

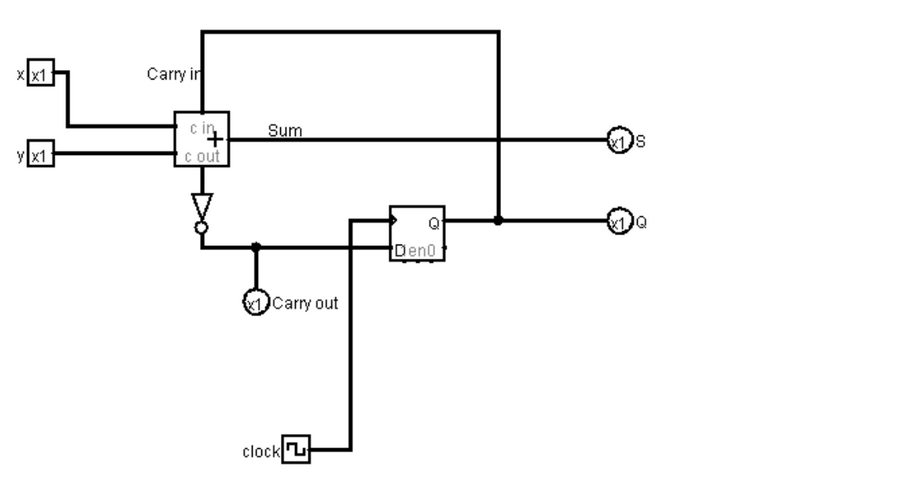
|  |  |
| --- | --- |
| C = xy + (x XOR y)z. | C = xy + yz + xz |
| C = xy + (x!y + !xy)z | C = xy + !xyz + x!yz (from table) |
| C = xy + x!yz + !xyz |  |
|  |  |
|  |  |

You can prove by deriving from  C = xy + (x XOR y)z to the answer you got in the question 3)

6) Based on 4) and 5), draw a circuit for the full-adder in Logisim simulator, **attach the image file and submit the circuit file**!

**Q2. (6 points) The following sequential circuit includes a full-adder (described in the previous question). Inputs are X, Y and carry-in, and outputs are the next state of S and Q.**

1) Implement the sequential circuit in Logisim simulator and **submit the circuit file**.



2) Complete the following truth table for the following sequential circuit: Note that the Carry out signal is an output, not the an input. The carry in signal is the same as the Q. You can change the Carry-in bit by clicking the D-FF.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| X | Y | Carry-in  (or Q before clock) | S (before clock) | Carry-out (before clock) | S (after clock) | Carry-out (after clock) |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 | 1 |

**Q3. (4 points) A sequential circuit has one D flip-flop and one JK flip-flop, two inputs x and y, and one output z. A is the output of D flip-flop, and B is the output of JK-flip-flop; A and B together form the "output state" of the circuit. The flip-flop *input* equations and the circuit output are as follows. Here DA is the D input of the D-flip flop of A, and JB, KB is the J and K input of the JK-flip flop of B.**

DA = ~xy + yB

JB = ~yB + xy

KB = xB + ~yA

z = x+~xy

1) Draw the logic diagram of the circuit and test it with Logisim.

**Please attach the circuit image and the generated table!**

**A screenshot of a cell phone

Description automatically generated**

x y A z B

0 0 0 0 0

0 1 0 1 0

1 0 0 1 0

1 1 0 1 0

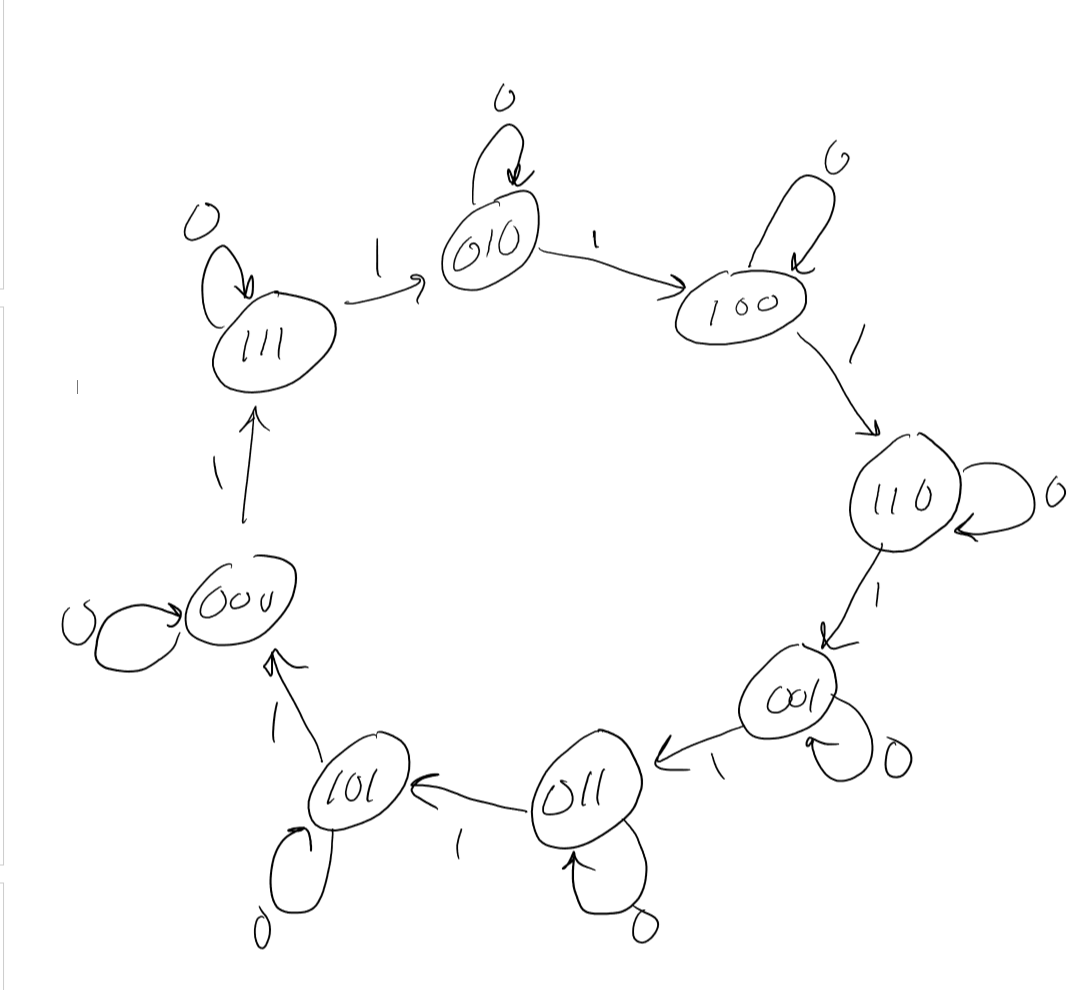
**Please also submit the circuit file!**  
2) Construct a state diagram of this circuit.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| x | y | A(t) | B(t) | z | A(t+1) | B(t+1) |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 |

**Q4. (10 points) Design a system with the following state changes: This is a sequential circuit with three flip-flops. The state sequence is changed with a clock as in the order of, 111, 010, 100,110, 001, 011, 101, 000, 111 and repeat. Use JK flip-flops.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A(t) | B(t) | C(t) | Clock | A(t+1) | B(t+1) | C(t+1) |
| 1 | 1 | 1 | 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 | 1 | 1 |

1) Draw a state diagram.



2) Construct an excitation table .

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A(t) | B(t) | C(t) | Clock(t) | A(t+1) | B(t+1) | C(t+1) | Ja | Ka | Jb | Kb | Jc | Kc |
| 1 | 1 | 1 | 1 | 0 | 1 | 0 | x | 1 | x | 0 | x | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | x | x | 1 | 0 | x |
| 1 | 0 | 0 | 1 | 1 | 1 | 0 | x | 0 | 1 | x | 0 | x |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 | x | 1 | x | 1 | 1 | x |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | x | 1 | x | x | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | x | x | 1 | x | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | x | 1 | 0 | x | x | 1 |
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | x | 1 | x | 1 | x |

3) Draw K-maps and derive Boolean equations using K-maps. Make the equations as simple as possible.

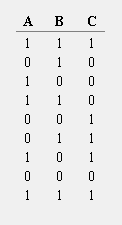
|  |  |  |
| --- | --- | --- |
| Ja | Ka | Jb |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | ab | !ab | !a!b | a!b | | c | x | 1 |  | x | | !c | x | 1 | 1 | x |   Ja = b + !c | |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | ab | !ab | !a!b | a!b | | c | 1 | x | x | 1 | | !c | 1 | x | x |  |   Ka = b + c | |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | ab | !ab | !a!b | a!b | | c | x | 0 | 1 |  | | !c | x | x | 1 | 1 |   Jb = !a + !c |
| Kb | Jc | Kc |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | ab | !ab | !a!b | a!b | | c |  | 1 | x | x | | !c | 1 | 1 | x | x |   Kb = !a + !c | |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | ab | !ab | !a!b | a!b | | c | x | x | x | x | | !c | 1 |  | 1 |  |   Jc = ab + !a!b | |  |  |  |  |  | | --- | --- | --- | --- | --- | |  | ab | !ab | !a!b | a!b | | c | 1 |  |  | 1 | | !c | x | x | x | x |   Kc = a |

4) Draw the system in Logisim simulator, attach the circuit image and submit the circuit file.

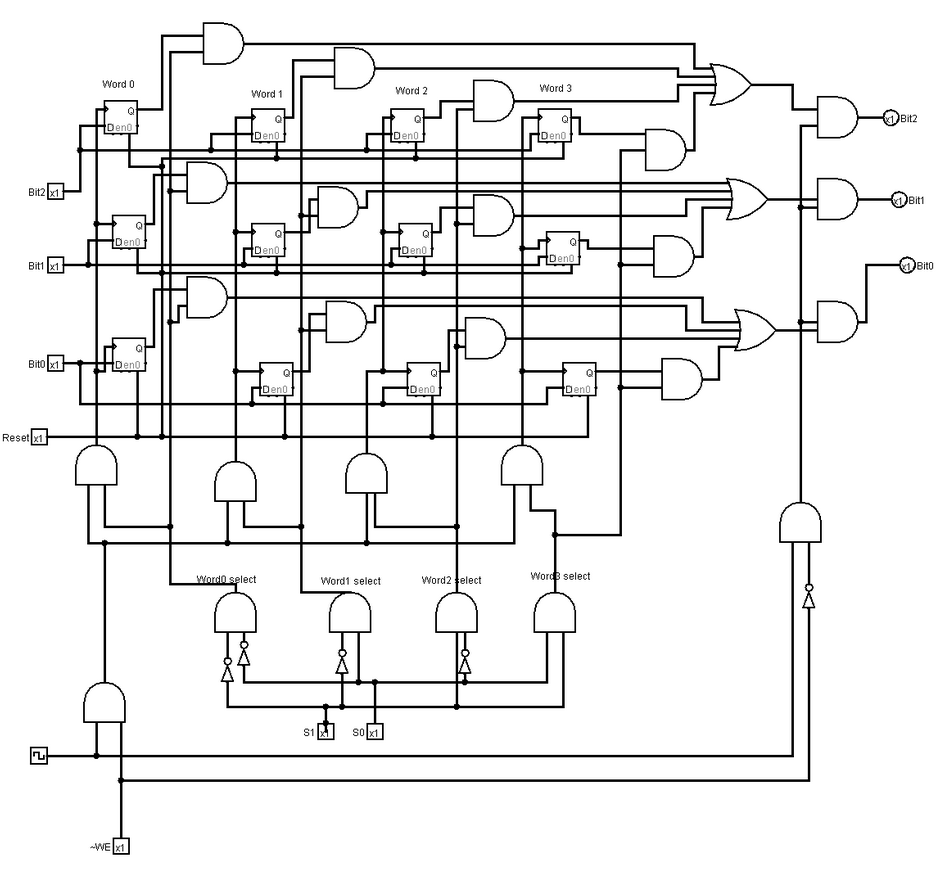
A close up of text on a black background

Description automatically generated

5) Test the system and **attach the generated table**.



**Q5.** **(2 points)** **The following circuit is a simple implementation for 4X3 memory chip. In this configuration, your memory chip has four addressable space and each data in the address is 3-bit long. In order to write/read a data into/from a specified address, you have to give a right signal to the address lines, S1 and S0.**



1) **(1 point)** Suppose you want to write a data 1 0 1 to the word 3 (address 3). How you will set the values in each case?

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **RESET** | **S1** | **S0** | **Bit2** | **Bit1** | **Bit0** | **~WE** |
| **0** | **1** | **1** | **1** | **0** | **1** | **0** |

2) (**1 point**) Suppose you want to read a data from the address 1. Give the correct values in each case. If some bits do not affect, then mark as X (don't care).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **RESET** | **S1** | **S0** | **Bit2** | **Bit1** | **Bit0** | **~WE** |
| **1** | **0** | **1** | **x** | **x** | **x** | **1** |