CS2271 MIDTERM

PART 1

You have been hired by a home security company to design and implement a home alarm system. The logic of the system is as follows: once the alarm system has been armed, it is to sound if the front door is opened, the back door is opened or either of two windows is opened (you can assume there are only two windows).

Ultimately, you will need to design the necessary circuit using Logisim to implement the situation described above. Your circuit should have five inputs (A = alarm, F = front door, B = back door, W1 = window 1 and W2 = window 2). A = 1 means the system is armed; A = 0 means it is disarmed. F = 1 means the front door is open; F = 0 means it is closed, and similarly for the back door and the windows. There should be one output, S. When S = 1 the alarm should sound; S = 0 means the alarm is silent.

Instructions

1. Construct a FSM diagram for the system. You may use either a Moore or a Mealy machine.
2. Build the truth table that represents the possible states of the system described above.
3. Write the functions expressing the truth table in sum-of-products form.
4. Simplify your functions if possible.
5. Build a circuit implementing your solution in Logisim. Your inputs and output should be labeled.
6. Submit your written answers to parts one through four above and your Logisim file for part 5.

PART 2

Design a hardware stack to hold a maximum of four values, each consisting of one bit. The stack has the following input lines:

- Clock
- POP
- PUSH
- input data

and a single output line. The output line always reports the current top of the stack.

(Recall that a stack pushes data onto the top of the stack, and pops data from the top of the stack. In other words, last in, first out. If more than four values are pushed on, older values are discarded.)

You may use AND, OR, XOR, and NOT gates, etc; flip-flops of any kind; and/or decoders, multiplexers, etc.

Optional hint: There is no need to keep track of how many items are on the stack. Just keep four D flip-flops for the up-to-four items on the stack. The initial values won't matter unless more things are popped off the stack than are pushed, and this is not allowed.

1. Construct a FSM diagram for the system. You may use either a Moore or a Mealy machine.
2. Build the truth table that represents the possible states of the system described above.
3. Write the functions expressing the truth table in sum-of-products form.
4. Simplify your functions if possible.
5. Build a circuit implementing your solution in Logisim. Your inputs and output should be labeled.
6. Submit your written answers to parts one through four above and your Logisim file for part 5.

PART 3

Design a hardware queue. This question looks rather similar to question five above, and I admit that it is quite similar in some ways; on the other hand, it has some notable differences.

The queue has the following input lines:

- Clock
- CLEAR
- ADD
- LEAVE
- input data

and a single output line. The output line always reports the current front of the queue. CLEAR is used to set the queue to empty.

(Recall that in a queue, data is added to the end of the queue, and data leaves from the front of the queue. In other words, first in, first out.)

Your queue will hold a maximum of two values, each consisting of one bit. If there are already two values in the queue and another is added, we don't care what happens, as this is a

prohibited action. Similarly, we don't care about the output value when there is an empty queue, nor about what happens once more LEAVEs than ADDs have been asserted since the last CLEAR; finally, you needn't be concerned with what will happen when your circuit is initially powered on, as the first operation will be a CLEAR, and you can assume that no more than one of the control lines will be asserted at a time. (These are just examples; in general, we don't care what happens if the queue is used incorrectly.)

You may use AND, OR, XOR, and NOT gates, etc; flip-flops of any kind; and/or decoders, multiplexers, etc.

Optional hint: In addition to the data-holding flip-flops, use two JK flip-flops to form a two-bit counter which keeps track of how many items are in the queue.

1. Construct a FSM diagram for the system. You may use either a Moore or a Mealy machine.
2. Build the truth table that represents the possible states of the system described above.
3. Write the functions expressing the truth table in sum-of-products form.
4. Simplify your functions if possible.
5. Build a circuit implementing your solution in Logisim. Your inputs and output should be labeled.
6. Submit your written answers to parts one through four above and your Logisim file for part 5.