

Name

CompSci/ECE350 Spring 2023 Midterm Examination 2 – 4/20/23

This is a closed book, closed notes, closed internet exam; you may consult 2 double-sided pages of notes that you should turn in with your exam. It is designed to take 75 minutes. Problem 2 carries double weight. Please answer all 4 questions. If you need to use extra pages for your answer, and your final answer is not where it is “supposed” to be on the exam paper, please be sure to say something like “see scratch page for final answer”.

**Please, if accurate, sign this statement before turning in your exam¹:
(or summarize on your own paper)**

I have neither given nor received any unauthorized help on this examination.

¹ If the statement is not accurate, we need to talk!

1. Minimization: Quine - McCluskey Algorithm

I used the website random.org, which uses atmospheric noise to generate truly random numbers, to generate the minterms for the function (so it may not be a “pretty” function!)
 $F(x_1, x_2, x_3, x_4) = \sum m(4, 6, 8, 9, 10, 11, 15) + d(7, 13)$.

[20 pts] Use the Quine-McCluskey algorithm to minimize this function: show all the lists used to identify prime implicants and the cover tables used to identify essential prime implicants; give a minimal form for the function and describe if it is unique or not and why.

Lists

Cover Tables

Minimal form and uniqueness

2. FSM Design

[40 points] Early in the e-commerce era (I am not making this up!), some MIT students designed a pizza-shaped Emergency Pizza Button which, when pressed, placed a pizza order with a cooperating local restaurant. Since there is just one button and not everyone wants the same pizza, they employ a Huffman code to encode desired ingredients, with more popular ingredients taking fewer button presses than less popular ones. Huffman codes have the property that, even though codewords are of different lengths, each can be unambiguously decoded. I presume the switch is a simple 2 position switch that can reliably send “0” or “1” (pressing the Pizza button down or up).



Given the following Huffman code:

00	extra cheese
10	pepperoni
010	sausage
011	pineapple
110	onions
111	EndOfPizza

a Pizza Emergency Code of 0010010111 would mean “send a pizza with extra cheese, pepperoni, and sausage;” note so also would 0100010111 as by construction all the shorter codes are distinct from all the longer codes so the sequence is unambiguous. The EndOfPizza code must be appended to the end of each message so we know when a pizza is complete. Sending the same code twice (or more) for the same pizza is ok, ..1010... will cause double pepperoni, and will have caused your FSM to fire the Pepperoni output twice.

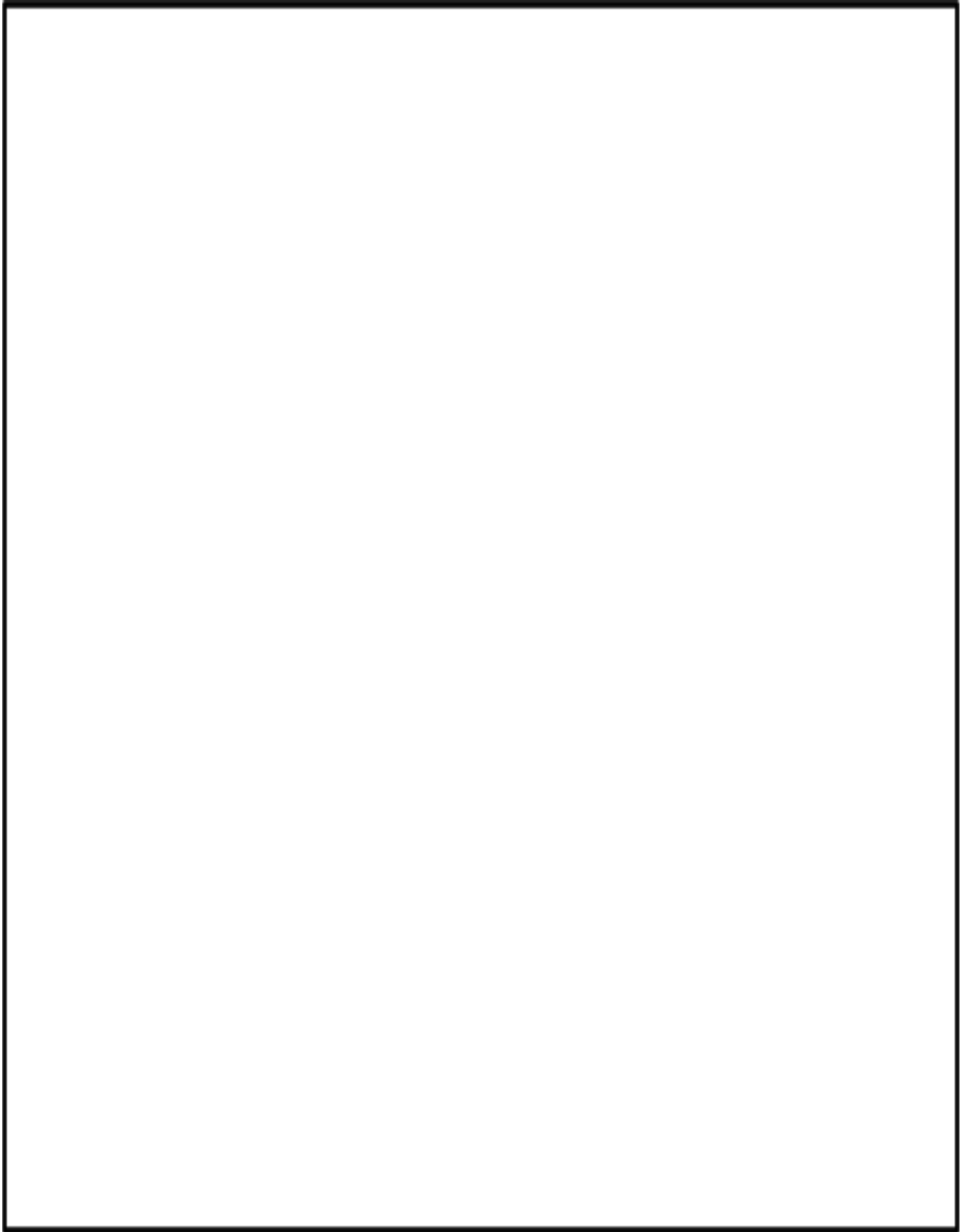
Design a **Moore-style** finite state machine to process incoming pizza emergency messages, signaling via output signals which ingredients to include on the pizza. (Note a pizza code of 111 would indicate a plain pizza with no special ingredients, i.e. no ingredient output signals active.) Hint: You will need 6 separate 1-bit outputs, one for each of the 5 possible ingredients, and “Pizza Done”.

For simplicity, you may assume the system's clocking "just works," each button press will generate a clock signal appropriately timed to allow the incoming 0 or 1 to be appropriately processed by your FSM.

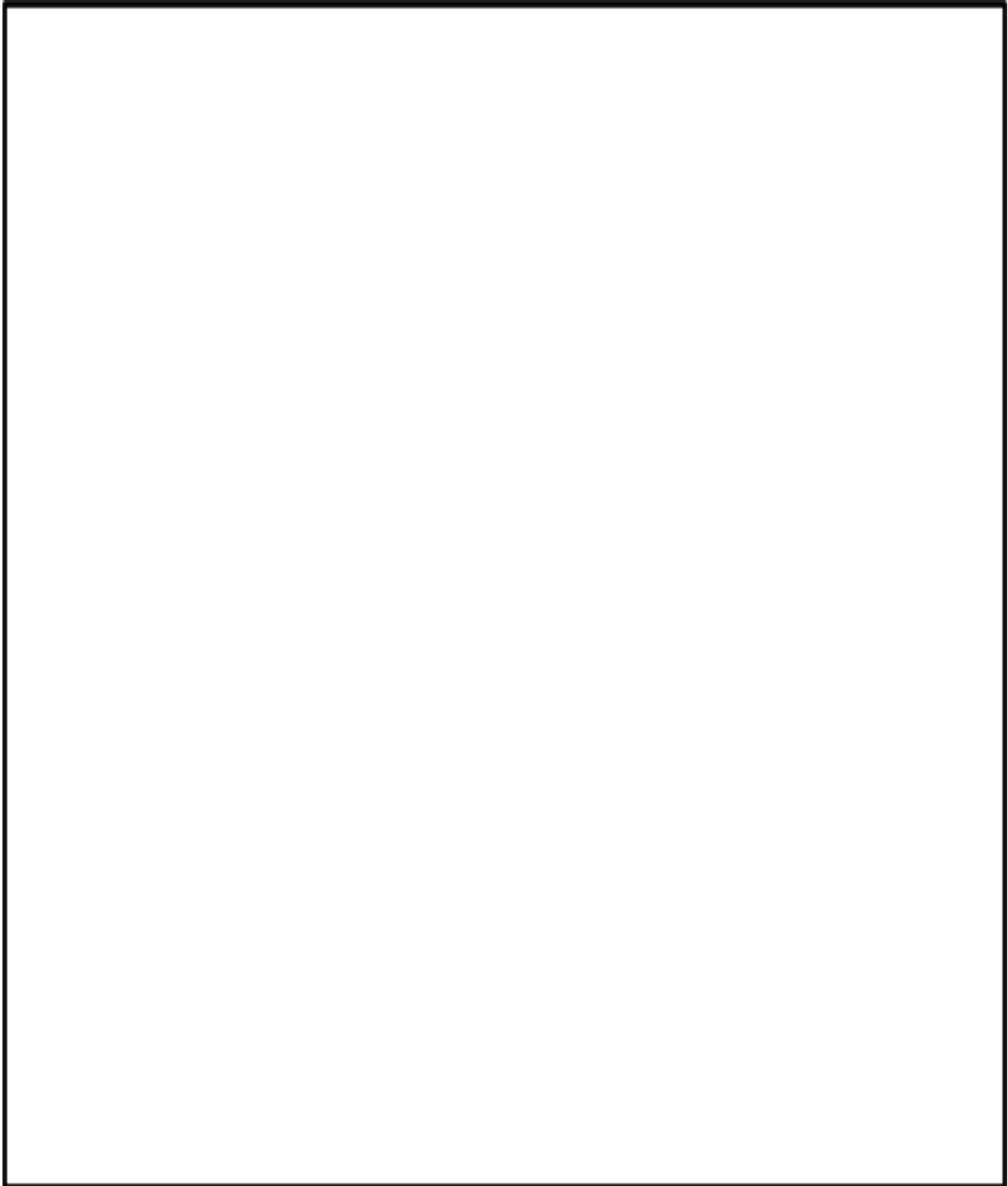
For your design, include

- a state diagram
- a state table
- use D flip-flops and a state encoding of your design to create the state transition table
- write down the (not necessarily minimized) logic equations for your flip-flops, the 5 ingredient outputs, and the "Pizza Done" buzzer; **you need NOT draw the resulting circuit!**

State Diagram (probably want to work on scratch paper first!)

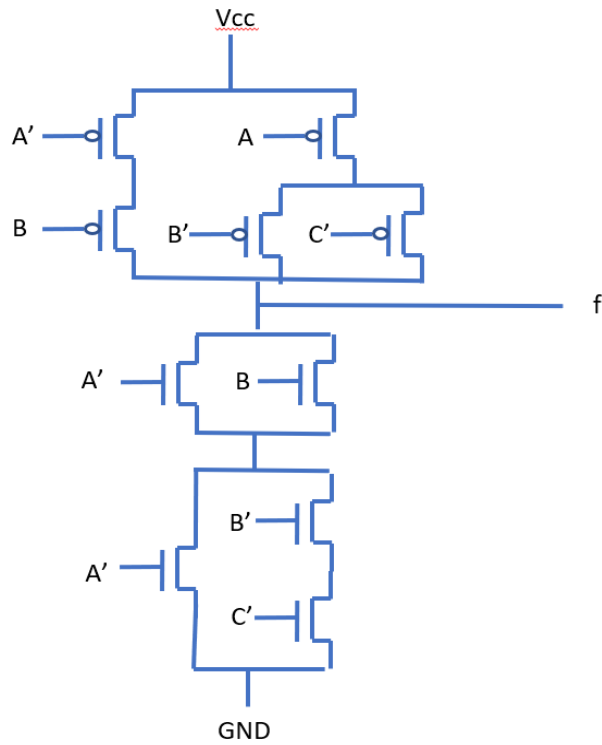


Equations (not minimized) for next state and the 6 output functions

A large, empty rectangular box with a black border, intended for the student to write the equations for the next state and the six output functions. The box is currently blank.

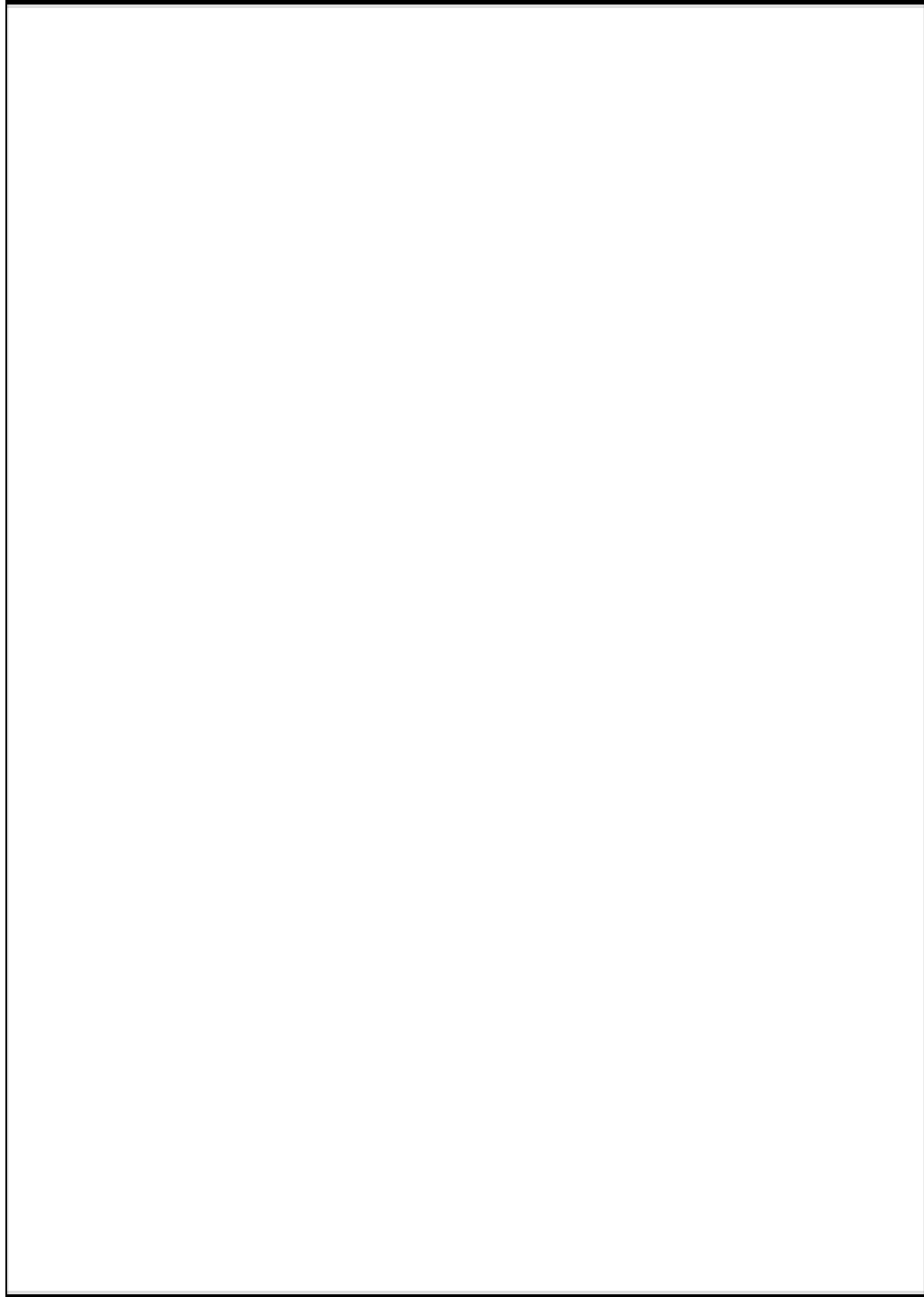
3. NMOS/PMOS/CMOS

- a. [8 pts] What function does the following CMOS circuit implement?



Function:

- b. [12 pts] Give an efficient CMOS implementation of $f=ab + ac + bc$. Derive the PUN and PDN, and draw the resulting circuit.



4. Hamming Codes

MOST OF THE CONFUSION ON THIS PROBLEM COMES FROM NOT PAYING CLOSE ATTENTION TO THE SECSSED ANSWER REQUESTED IN PART a, AND THE SECDED ANSWER REQUESTED IN PART b.

[20 pts] It took awhile before the computer industry standardized on 8-bit bytes; some early machines even used odd numbers of data bits. Let's build a SECSSED memory system that protects 15-bit data words.

- a) [4 pts] First determine how many Hamming bits you need for the base **SECSSED** (NOT SECDED!) Hamming code for 15 bit data words

Show the **SECDED(!!)**-encoded versions of original data words

- b) [8 pts] 101011110100101
c) [8 pts] 010011101011000

A: Number of bits for SECSSED (NOT SECDED) with 15 information bits

<BLANK HAMMING TEMPLATE ON NEXT PAGE FOR b AND c>

