Midterm exam (self-timed, take-home, **trust-based** exam), CS 110, Spring 2022.
Due date: April 6th, Noon!
Student Name and ID:
Max Time: 180 mins (3 hours, including download, scan, upload, will be monitored by the time taken to accept and upload–make sure you practice the upload part. If you *cannot* upload it for some reason, please send me an email with the subject midterm1_cs110. The emailed copies will NOT be graded; they are just for a backup. Only those submissions that are submitted to GitHub shall be graded.)
Maximum points: 100 (25% of the total grade)

- Please write your name and Student Id
- There are three parts, and the number of questions to be attempted is indicated in each part.
- **How should you use the allotted time?**
    - Scan the whole question paper once. While reading the question paper, you can mark questions as E or M, or D. E for those questions that you think are easy, M for those that you think are moderately difficult, and D for those questions you think are difficult.
    - Start solving questions that you have marked E followed by M and D.
    - Almost always, students think that the question is easy and is done, and they submit the answer sheet. Later they regret that they should have verified their answers once again. Therefore, I recommend you verify and correct the answers instead of submitting the exam early.
- **Where to write your answers?**
    - There is sufficient space for your answer right after the question. However, you can use extra pages and attach the same at appropriate places [insert the answers right after the corresponding questions].
- **Resources that you can use?**
    - It's a closed book, closed notes, and a closed computing devices exam. You can, however, use a calculator, and a handwritten A4-size cheat sheet (both sides) is allowed, and the cheat sheet should be submitted along with the answers sheets.
    - Alarm the clock to time yourself.
    - If you have a letter of accommodation, consider the flexibility granted.
- **Partial grading?**
    - Make sure you write the as accurate output as possible. Good answers shall be rewarded
    - Some questions may have multiple subquestions. Make sure you answer all parts of the question. If you answer only partial questions, partial grades shall be awarded
    - I focus on the process, so if you dont know the complete answer. Write down your approach in simple English.
- **How to submit?**
    - Once you are done, please take pictures to convert them to a pdf (you can use Android or iOS apps to do that automatically) and then upload them to the corresponding repository on your GitHub as soon as possible.
    - The time spent printing, scanning, and uploading SHALL be counted.
- **Opportunity to ask questions?**
    - Since this is a take-home exam, I cannot make myself available round the clock. Therefore, there would be no opportunity for asking questions or clarifications. If there is any confusion, please assume and write your assumptions and solve the question based on the assumption.
- I declare that I will not share this exam with anyone at any point.

------Sign here----

Part1

Answer any SIX. Provide examples wherever necessary. (6x6 = 36 points)

1. The truth table shown below is supposed to reflect the function F(x, y, z) = y(x + z'), but two lines are in error. Identify them. **[6 points]**

   A) 1 and 2

   B) 3 and 4

   C) 5 and 6

   D) 7 and 8

#	*x*	*y*	*z*	*F*
1	0	0	0	0
2	0	0	1	0
3	0	1	0	0
4	0	1	1	1
5	1	0	0	0
6	1	0	1	0
7	1	1	0	1
8	1	1	1	1

2. Write short notes on Von Neumann architecture vs. Harvard architecture **[6 points]**

3. You are given an array/list L of n + 1 integers, ranging between 1 and n. All numbers appear exactly once, except one number, which is duplicated. Write a program to find the duplicated number. You are *not allowed* to use any arithmetic operators. (Hint: the solution uses XOR)

[6 points]

4. What is the simplified expression for the following Kmap? Show your steps, and verify your answer by simplifying the expression using Boolean identities? **[6 points]**

X \ YZ	00	01	11	10
0	1	1	1	0
1	1	1	0	0

A) Z + X'YZ

B) Y' + X'YZ

C) Y' + X'Z

D) Z + X'Z

5. Explain *three* ways of representing signed numbers in computers. List the advantages and disadvantages of each of them. Which one do we use predominantly and why? **[6 points]**

6. Write a short note on the Booth algorithm vs. Standard multiplication algorithm for signed integers. Provide examples of the *best* and *worst* cases for Booth and Standard multiplication algorithm **[6 points]**

7. How are fractional numbers represented in computers? Why do we normalize them before storing them in the computers? Explain your answer with example(s). **[6 points]**

8. Describe two algebraic identities that may or may not hold in floating-point representation. Explain why? with examples? **[6 points]**

Part II

Attempt any FIVE from the following: (8 x 5 = 40 points)

1. Convert the following decimal fractions to binary with a maximum of six places to the right of the binary point: **[8 points]**

    a) 25.84375

    b) 57.55

    c) 80.90625

    d) 84.874023

2. Convert the following binary fractions to decimal: **[8 points]**

    a) 100001.111

    b) 111111.10011

    c) 1001100.1011

    d) 10001001.0111

3. Which of the following equations correctly reflects the truth table below? What would be the simplified version of the correct equation **[8 points]**

x	*y*	*z*	*F*
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

A) (x'y'z + x'yz' + xy'z + xyz)'
B) xy'z + x'yz' + xy'z + xyz
C) (x'y'z' + x'yz + xy'z' + xyz')'
D) x'y'z' + xvyz + xy´z' + xyz'

4. Perform multiplication of 0110 and 0010 using Both Standard and Booth's algorithms (see Figure 10.9 and 10.12), assuming signed two's complement. Show all the steps. **[8 points]**

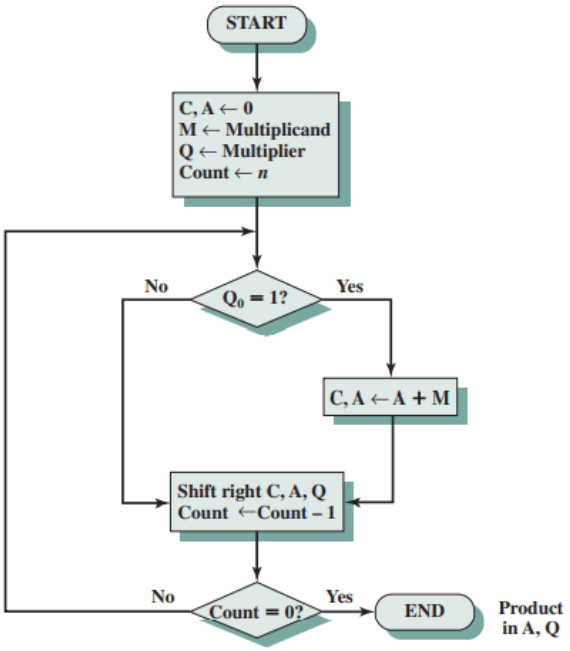

Figure 10.9 Flowchart for Unsigned Binary Multiplication

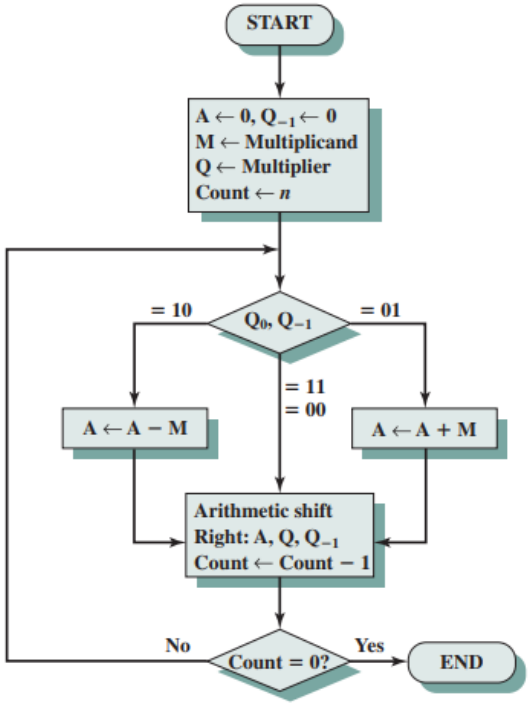

Figure 10.12 Booth's Algorithm for Twos Complement Multiplication

5. Assume we are using the simple model for floating-point representation shown in Figure 2.2, i.e. the representation uses a 14-bit format, 5 bits for the exponent with a bias of 15, a normalized mantissa of 8 bits, and a single sign bit for the number: **[8 points]**

    a) Show how the computer would represent the numbers 100.0 and 0.25 using this floating-point format.

    b) Show how the computer would add the two floating-point numbers in part a by changing one of the numbers so they are both expressed using the same power of 2.

    c) Show how the computer would represent the sum in part b using the given floating-point representation. What decimal value for the sum is the computer actually storing? Explain.

FIGURE 2.2 Floating-Point Representation

6. Let a = 1.0×2^{9}, b = -1.0×2^{9}, and c = 1.0×2^{1}. Using the floating-point model described in the text (the representation uses a 14-bit format, 5 bits for the exponent with a bias of 15, a normalized mantissa of 8 bits, and a single sign bit for the number), perform the following calculations, paying close attention to the order of operations. What can you say about the algebraic properties of floating-point arithmetic in our finite model? Do you think this algebraic *anomaly* holds under multiplication as well as addition? **[8 points]**

   b + (a + c) =

   (b + a) + c =

7. Using DeMorgan's Law, write an expression for the complement of F if

   a. F(w,x,y,z) = xz'(x'yz + x) + y(w'z + x'). **[8 points]**

Part III

Write the Boolean expression for Play=yes (yes = 1, no = 0), simplify the expression using both Boolean identities, and verify using KMap. Considering that some columns have more than two values, we must discretize them into two (binary) values. Other than the given combinations, the rest of the combinations could be considered "dont care," which means you can consider the dont care cases as ZERO or ONES, whichever simplifies makes a bigger group –better simplification while solving them using kmap. List how many gates will be needed and which ones in total to implement the original boolean expression and the simplified ones. **[24 points]**

Use the following rules to do so:

Temperature: if <=75: GOOD, else: BAD

Humidity: if <=80: GOOD, else: BAD

Outlook	Temperature	Humidity	Windy	Play
sunny	85	85	FALSE	no
sunny	80	90	TRUE	no
rainy	70	96	FALSE	yes
rainy	68	80	FALSE	yes
rainy	65	70	TRUE	no
sunny	72	95	FALSE	no
sunny	69	70	FALSE	yes
rainy	75	80	FALSE	yes
sunny	75	70	TRUE	yes
rainy	71	91	TRUE	no

PS: Check your answers before submitting.

How to upload?

Once you are done, take pictures, turn them into a pdf (there are apps that do it automatically, installing them will save you time), and then open your Github account and upload the file as depicted below: