CS 151
Quiz 4

Name                      : __________________ , __________________
                     (Last Name)                 (First Name)

Student ID               : _______________

Signature                 : ________________

Instructions:

1. Please verify that your paper contains 9 pages including this cover.
2. Write down your Student-Id on the top of each page of this quiz.
3. This exam is closed book. No notes or other materials are permitted.
4. Total credits of this quiz are 70 points.
5. To receive credit you must show your work clearly.
6. Calculators are NOT allowed.
Q1 [Memory Design] [15 points]

Design a 24K*64-bit RAM using standard 8K*16-bit RAM modules like the one shown below.

Use a minimum number of the following logic components in your design (no other components may be used for this design):

1) Priority Encoder
2) Decoder
3) Multiplexer

You don’t need to connect all input lines (data, addr, rw, en), however, you DO need to label the bit ranges for each data field, and address field as well as the enable signals.
<This page is intentionally left blank>
Q2 [Critical Path and Frequency Calculation] [15 points]

Shown below is a simple filter composed of 3 input registers (A, B, C) and an output register (O), a series of adders, multipliers and a multiplexer with two select signals.

2a. Determine the register-to-register critical path and the longest path delay for the circuit. Draw a circle around every component in the critical path. [10 points]

Assume no wire delay, and the following component delays:
- Multipliers take 5 ns
- Adders take 2 ns
- Multiplexers take 1 ns
2b. Based on the longest path delay obtained from part 2a, what should the frequency of this circuit be? (Make sure your answer is in GHz, MHz, KHz, or Hz) [5 points]
Q3 [RTL Design-C-code to Gates] [40 points]

The following C-code tries to count the number of unsigned byte numbers divisible by four from array A.

Inputs: byte A[128], bit start
Outputs: byte mod4_count, bit done

```c
main() {
    short uint i, count, data;
    while(1) {
        while(!start); // wait for start signal
        done = 0;
        i=0;
        count = 0;
        data = 0;
        mod4_count = 0;

        while(i<128) {
            data = A[i];
            if(data%4==0) {
                count = count + 1;
            }
            i = i + 1;
        }

        mod4_count = count;
        done=1;
    }
}
```

3a. Using the templates shown below, convert EACH of the C-code statements (shown in bold) to its high level state machine representation. You should start from the outermost loop. [30 points]
<This page is left blank intentionally>
<This page is left blank intentionally>
3b. Now that you have transformed the C-code statements to their high level state machine representation, merge them together to represent the final high level state machine. Make sure you remove all unnecessary states. [10 points]