Jetic Gū

Columbia College

1. Handwritten submissions and proprietary formats (e.g. Pages or MS Word) **will not be graded**.
2. Late submission and resubmission policies are stated on the course webpage.
3. Mathematical expressions must be written entirely using LaTeX, otherwise **50%-100%** of marks will be deducted.
4. Circuits must be **tested** using appropriate IO against a truth table/specification. Untested circuits will receive 0.

Submission File structure:

```
submission.zip
    — c1-1.cct
    — c1-2.cct
    — c1-3.cct
    — c2.cct
    — c3.cct
    — c4.cct
    — c5-1.cct
    — c5-2.cct
    — c6.cct
    — c7.cct
    — lib.clf
```

All circuit files are 1pt each.

Lab 2

First Half

1. Create the following circuit components. Your CCT file **must contain** the following component being tested using **switches/probs/hex keyboard/hex display** after being saved as a component in your library:
    1. 4-to-16 Decoder implemented using a 3-to-8 decoder and 1-to-2 decoder (`c1-1.cct`);
       Requirement: you must implement the 3-to-8 decoder on your own
    2. 8-to-3 priority encoder with validity bit (`c1-2.cct`);
       Requirement: your CCT file must show the component tested using a 3-to-8 decoder and HEX keyboard and HEX display
    3. 4 channel 4bit Multiplexer implemented using the 4 channel 1bit Multiplexers (`c1-3.cct`);
       Requirement: your CCT file must show the component being tested using HEX keyboards and HEX displays

2. Implement the following Boolean function with a 3-to-8 decoder:

   $$F(A, B, C) = \Sigma m(1,2,3,6)$$

   You must ONLY a the decoder and an additional OR gate. Save the circuit as `c2.cct` .

3. Implement a 4-bit binary plus 1 incrementer (`c3.cct`). Your CCT file **must contain** the following component being tested using **switches/probs/hex keyboard/hex display** after being saved as a component in your library. You should consider using the 5-step systematic design procedure.
   Requirement: your CCT file must show the component being tested using HEX keyboards and HEX displays. You must **NOT** use full adders to implement this.

4. A parity checker is a component that verifies the parity of the entire input. For example, an 8bit even parity checker would be able to tell that 10010001 contains error, while 00011000 does not. Given 8bit input $A_{7:0}$, design an even parity checker that outputs 0 when no error is found, and 1 when there is. Save the circuit as `c4.cct` . Your CCT file must show the component being tested using 2 hex keyboards and 1 binary prob.

Second Half

5. Create the following circuit components. Your CCT file **must contain** the following component being tested using **switches/probs/hex keyboard/hex display** after being saved as a component in your library:
   1. 4-bit binary adder with $X_{3:0}$, $Y_{3:0}$, Z as input, $S_{3:0}$ and C as output (`c5-1.cct`);
      Requirement: your CCT file must show the component being tested using HEX keyboards and HEX displays
   2. 4-bit binary adder-subtractor with XOR and a 4-bit Adder, the component should use $X_{3:0}$, $Y_{3:0}$, Sub as input, $S_{3:0}$ and C as output (`c5-2.cct`);
      Requirement: your CCT file must show the component being tested using HEX keyboards and HEX displays

6. Implement a 4-bit unsigned selective 2s complementer. It should take 4bits $A = A_3 \ldots A_0$ and C as input, $D = D_3 \ldots D_0$ as output. It should output A when $C = 0$, the unsigned 2s complement of A when $C = 1$. Save the circuit as `c6.cct` . (Hint: use your 4-bit binary plus 1 incrementer)

7. Use 5 step systematic design procedure to design a 2bit by 2bit multiplier. It should take $X_{1:0}$, $Y_{1:0}$ as input, $P_{1:0}$ and O as output where O is an overflow bit. Save the circuit as `c7.cct` . Your CCT file must show the component being tested using hex keyboards and displays as well as binary probs.