ENGR xD52: HW b010

Due October 7th 11PM EST

This homework is to be done primarily alone. If you get stuck, you may consult anyone you like after putting in real effort. Annotate collaboration per problem. This should be **typed** and emailed to CompArch13@gmail.com

Expected time is less than one hour.

Look Up Tables

This section examines how a gate is represented on a piece of FPGA fabric. Use a 3 input Look Up Table to recreate the given logic function.

For each, do the following:

1. Write the truth table that will be written to a 3LUT to implement the logic.
2. Draw the corresponding circuit if it is useful for you. **Do not turn this portion in.**
3. Write the corresponding unsimplified Boolean equation. Note – this should start out obnoxiously long and be in "Sum of Products" Notation.
4. Use Boolean laws to simplify the Boolean equation back down to the original function. Show all work.

1.1 Two Input OR Gate (A+B)

1.2 The Carry Bit of a base 2 Full Adder (AB+AC+BC)

1.3 A 2 input Mux without Enable. Map the input 1 to A, input 2 to B, Select to C.

A	B	C	A+B	AB+AC+BC	2Mux
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

Estimation

This is intended as a “back of the envelope” estimation. This problem is intentionally missing information. Do not use any outside sources or people to fill in these gaps. Identify what it is missing, make best guess assumptions and document them.

2.1 How long would it take to exhaustively test the 32-bit “Add” instruction on your laptop? That is to say that it tries all possible combinations of {32-bit number}+{32-bit number}.

2.2 How much would the electricity cost for this exhaustive test?

Specification

Pretend that you are a midlevel group manager of a silicon company in the mid 80s. Select appropriate music/attire for this exercise. Create a specification for an ALU that performs the following functions: AND, OR, Add, Subtract, Multiply.

This specification’s intended audience is competent engineers that will incorporate your team’s ALU into their larger design. It should clearly indicate inputs, outputs, behaviors, etc., but it should **not** contain any implementation detail.