**PRE-LAB QUESTIONS OR EXERCISES**

**Part 1 Questions:**

1. If you were to make a complete (un-abbreviated) truth table for your ALU, how many rows would it need? (The answer should tell you why I did not want a simulation including ALL input combinations.)
   1. It Would take 2048 rows because there are 11 inputs.
2. What changes would be necessary to the design already made if you wanted to add two more functions, F=A NOR B and F=A XOR B, where XOR is exclusive or?
   1. You would need 6 input Mux’s instead of the 4 input ones we are using to accommodate the added inputs.

**Part 2 Questions:**

1. Draw the single simple device that can be added to your circuit design to “remember” the last carry output. Specify the inputs and outputs for this device.
2. Will a divide by two work for all 4-bit 2’s complement numbers? Explain.
   1. No, if you try to divide 7, you ill get 3.
3. Describe how you can take the 2’s complement of a number, i.e., if A is loaded with a number, get the 2’s complement of A into B.
   1. You can complement and add 0001 to that number.
4. Describe how you subtract with your CPU. Hint: See above question.
   1. Take the 2’s complement and add that to the number you re trying to subtract from.
5. Suppose you’re not allowed to use a flip-flop that has an asynchronous CLR or SET, how can you add a function that clears the contents of either A or B?
   1. Set all values of the inputs to 0, this will set REGA and REGB to all zeroes.

**PROBLEMS ENCOUNTERED**

N/A

**REQUIREMENTS NOT MET**

N/A

**FUTURE WORK/APPLICATIONS**

In this lab we learned how to make a simple CPU, which on itself can be applied ad used for various functions. By learning using this basic design we can build on it and if we wanted, add even ore functions, applications and abilities. CPUs are staples in almost every electronic device, and it is valuable to understand how one works and operates.

**PRE-LAB REQUIREMENTS (Design, Schematic, ASM Chart, VHDL, etc.)**

**Part 1:**

**A close up of a map

Description automatically generated**

**A screenshot of a video game

Description automatically generated**

**Part 2:**

**A close up of text on a white background

Description automatically generated**

**A screenshot of a cell phone

Description automatically generated**

A screenshot of text

Description automatically generated

**Individual Pictures Below:**

**A screenshot of a cell phone

Description automatically generated**

**A screenshot of text

Description automatically generated**

**A screenshot of a cell phone

Description automatically generated**

**A close up of a map

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A close up of a map

Description automatically generated**

Displays (REGA or REGB) to the output

Displays the REGA shifted right one bit to the output

Displays REGA shifted left one bit to the output

Displays (REGA + REGB) to the output

Displays (REGA and REGB) to the output

Displays the REGA to the output

Displays the REGB to the output

Displays the complement of REGA to the output

**A screenshot of a cell phone

Description automatically generated**

**Simple Functions:**

**A.**

**A screenshot of a map

Description automatically generated**

The second clock stores the complement of REGA into REGB

The first clock loads REGA with value 0001

**B.**

**A screenshot of a computer

Description automatically generated**

The third clock ANDS (REGA and REGB) and stores it in A

The second clock loads REGB with value 1001

The first clock loads REGA with value 0110

**C.**

**A screenshot of a computer

Description automatically generated**

The third clock ORS (REGA or REGB) and stores it in B

The first clock loads REGA with value 0110

The second clock loads REGB with value 1001

**D.**

**A screenshot of a computer

Description automatically generated**

The second clock SUMS (REGB + REGA) and stores it in A

The first clock loads REGA with value 0001

**E.**

**A screenshot of a computer

Description automatically generated**

The first clock loads REGA with value 1101

The second clock SHIFTS REGA left once and stores it I B

**F.**

**A close up of a map

Description automatically generated**