DALHOUSIE UNIVERSITY
Department of Electrical & Computer Engineering
Digital Circuits - ECED 2200

Experiment 2 - Arithmetic Elements

Objectives:
Using **Logisim:**
1. To implement a Half adder circuit
2. To implement a Full adder circuit
3. To design and implement a Half subtractor circuit
4. To design and implement a Full subtractor circuit
5. To implement a circuit using Chained adders
6. To design and implement a circuit using Chained subtractors

Theory:
A note on numbering systems:

In decimal, numbers are expressed in ones place, tens place, hundredths place, and so on. For instance the decimal number 137 has a ‘7’ in the ‘ones’ place, a ‘3’ in the ‘tens’ place, and a ‘1’ in the ‘hundredths’ place. This is a base ten system, because each placeholder is a factor of ten greater than the previous. (1, 10, 100…).

Binary is a base two system. This means that each placeholder is a factor of two greater than the previous. For instance 1110 has a ‘0’ in the ‘ones’ place, a ‘1’ in the ‘twos’ place, a ‘1’ in the ‘fours’ place, and a ‘1’ in the ‘eights’ place. To convert this number to decimal, add together the value of all the placeholders that have ‘1’ in them instead of zero. For example: 1011 binary would be 8+2+1=11 decimal, while 1110 would be 8+4+2=14 decimal. In this lab we will only be working with binary numbers with one, two, or three placeholders: 0, 1, 10, 11, 100, 101, 110, 111 (0,1,2,3,4,5,6,7 decimal).

How to add and subtract in binary:
Binary addition and subtraction is very similar to decimal addition and subtraction. For instance, in **decimal** addition and subtraction carries and borrows are used:

```
 14                                        24
+ 8   ← Carry has occurred (4+8=12        - 8   ← Borrow has occurred
 22          The tens place in ‘12’ is     16          (4 must borrow
             carried over to next column)              10 from tens place
                                                       giving 14-8=6)
```

Similarly, **binary** addition and subtraction involves carries and borrows:

```
 01                                        10
+01   ← Carry has occurred (1+1=2 so      -01   ← Borrow has occurred
 10          The ‘1’ is carried to the     01          (0 must borrow 2
             ‘twos’ place giving 10)                   from ‘twos’ place
```

giving 2-1=1)

Note that these operations involve two numbers with two places each. (The numbers being operated on are each 2-places).

Our goal in this lab is to implement addition and subtraction using logic gates. In the half adder and half subtractor, the inputs (switch 1 & 2) represent the two numbers we are adding or subtracting. The output LEDs show the resulting **sum** (adder) or **difference** (subtractor) and whether a **carry** (adder) or **borrow** (subtractor) has occurred.

Procedure:

Part 1. Half Adder

Look at the half adder circuit diagram attached to the lab. Switches **S1** and **S2** can either be opened or closed (0 or 1). In the half adder device we add them together (S1 + S2) with the result shown on LED1 (**sum**) and LED2 (**carry**). The half adder is limited in that it can only add two single-place numbers. With a half adder 0+0, 1+0, 0+1 or 1+1 are the only options (two 'ones' place numbers) → 10+01 or 01+11 is not possible because the numbers involved have two places/digits instead of one place/digit (a single adder cannot accept a number with a 'twos' place).

a) Using the skills learned in Lab 1, construct the Half Adder as shown on the attached schematic.
b) Simulate the behavior model of the half adder. Vary the two inputs and construct a truth table showing your results.

Part 2. Full Adder

The full adder is composed of two half adders chained together. The first half adder's **sum** output becomes one of the inputs for the next half adder stage. Examine the full adder's circuit diagram and you will see that it contains two half adders. The full adder by itself is still limited to adding only single-place numbers (no 'twos' places), but now an additional switch, switch 3 (**S3**) is available. S3 is used to bring forward a **carry** ('twos' place) from a previous stage of logic: To add multi-place numbers, full adders are chained together (cascaded) and the **carry** output (LED2) from one full (or half) adder is tied to S3 of a second full adder. Each additional full adder lets you have an additional place in the two numbers you are adding together. For instance if you chain a half adder to a full adder you can now add two, 2-place numbers together (i.e. 10+11). If you chain a half adder and two full adders together, you can add two 3-place numbers together (i.e. 100+101). We will play with this more in part five of the lab.

a) Construct the Full Adder as shown on the attached schematic.
b) Simulate the Full Adder: vary the three inputs and construct a truth table showing your results.
c) How is the Full Adder different from the Half Adder?

Part 3. Half Subtractor

The half subtractor and full subtractor work much the same way as their adder counterparts except we are subtracting the numbers instead of adding them. In the half subtractor we have two single place inputs, **S1** and **S2** that are subtracted from each other (S1-S2). The outputs (LED1 & 2) show the result (**difference**) and whether or not a 'borrow' occurred (**borrow**).

a) Design a Half Subtractor using a truth table and a Karnaugh map. Construct the Half Subtractor using Logisim.
b) Simulate the Half Subtractor: vary the inputs and construct a truth table showing your results.
c) What effect does a negative result have on the outputs? (0 – 1 = -1)

Part 4. Full Subtractor

The full subtractor is composed of two half subtractors chained together. The '**difference**' output of the first half subtractor becomes one of the inputs for the second half subtractor. Again, note the third switch, **S3**. This switch is used to tell the logic that a 'borrow' occurred from the previous stage when chaining many full subtractors together. The '**final borrow**' output of the previous stage is tied to **S3** of the next stage. We will look at this in more detail in part six of the lab.

a) Design a Full Subtractor using two Half Subtractors. Construct the Full Subtractor using Logisim.
b) Simulate the Full Subtractor: vary the inputs and construct a truth table showing your results.
c) What effect does the 'borrow' input (**S3**) have on the circuit?

Part 5. Chained Adders

On their own, single full/half adders are very limited. They can only add two single-place numbers together. Chaining (also called cascading) half and full adders together allow larger numbers to be added. The number of adders used is the maximum number of places that the circuit can handle (e.g. two adders allow addition of two-place numbers, six adders allow addition of six-place numbers, etc.)

a) Consult the 'Chained Half and Full-Adders' diagram and use it as a guide for combining the half and full adder circuits you created in parts 1 & 2.
b) Simulate the Chained Adders: vary the inputs and construct a truth table showing your results.
c) What effect do the four inputs have on each output? Hint:

(S3)(S1) ← First number
+(S4)(S2) ← Second number
(X1)(X2)(X3) ← Result

d) Why is the first element in the chained adder a half adder instead of a full adder?
e) What would you have to do if only full adders were available for the circuit?

f) Draw a simple block diagram (like the one you used in part ‘a’ showing how you would create an Adder capable of adding two four-place binary numbers. Create a formula like the one given for the hint in part ‘c’ showing the relationship between the different inputs and outputs, referenced to your diagram. (i.e. if you label an output X1 on your diagram, write it as X1 in the formula)

Part 6. Chained Subtractors

The subtractor circuits have similar limitations to the adders. On their own, they can only subtract single-place numbers. Like the adders, they can be cascaded to handle larger numbers. Successive levels of cascading give the ability to subtract larger numbers.

a) Construct a 4-bit chained subtractor. It will be very similar to the chained adder circuit you constructed in the previous part, but with two 4-bit inputs.
b) What happens when the result is a negative number?

Some Useful Terms:

You might find these terms handy when you are writing your report:

Bit: A single-place binary number (a single ‘1’ or ‘0’ is a Bit)

Byte: A binary number containing eight Bits (e.g. 11011110, or 10101010, etc.)

Nibble: Half a Byte, or four Bits (1100, 1010, etc)

Note that the adder you design in part 5e adds two Nibbles together

Augend: The first number in an addition

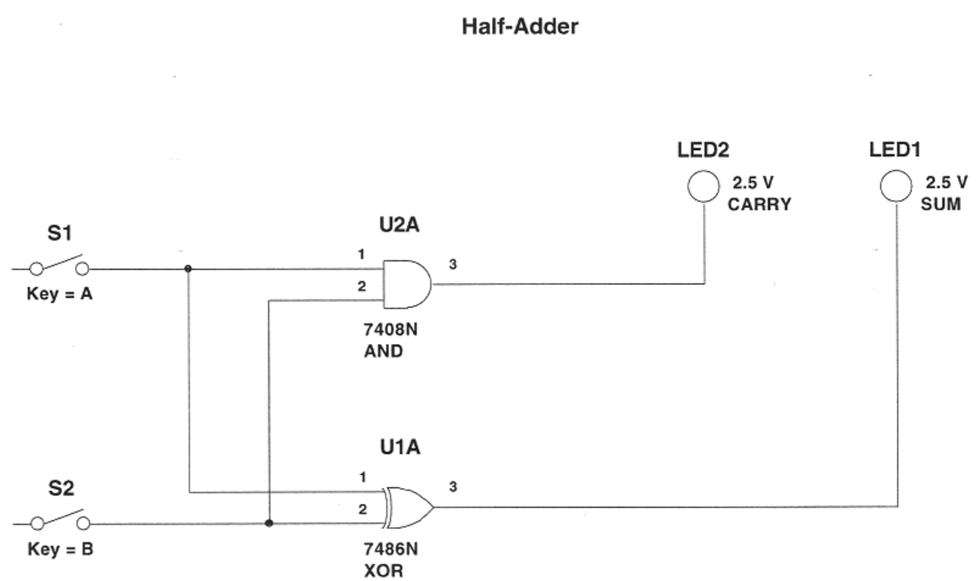
Addend: The second number in an addition

Sum: The result of the Augend + Addend

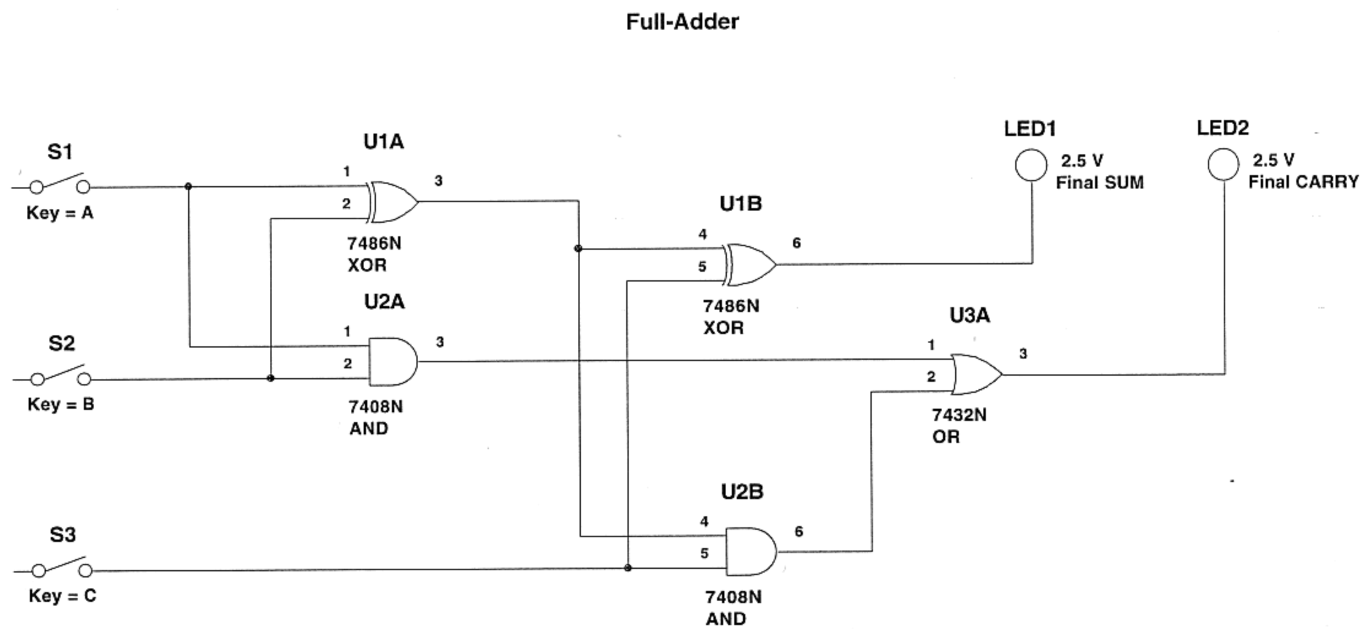
Minuend: The first number in a subtraction

Subtrahend: The second number in a subtraction

Difference: The result of the Minuend – Subtrahend

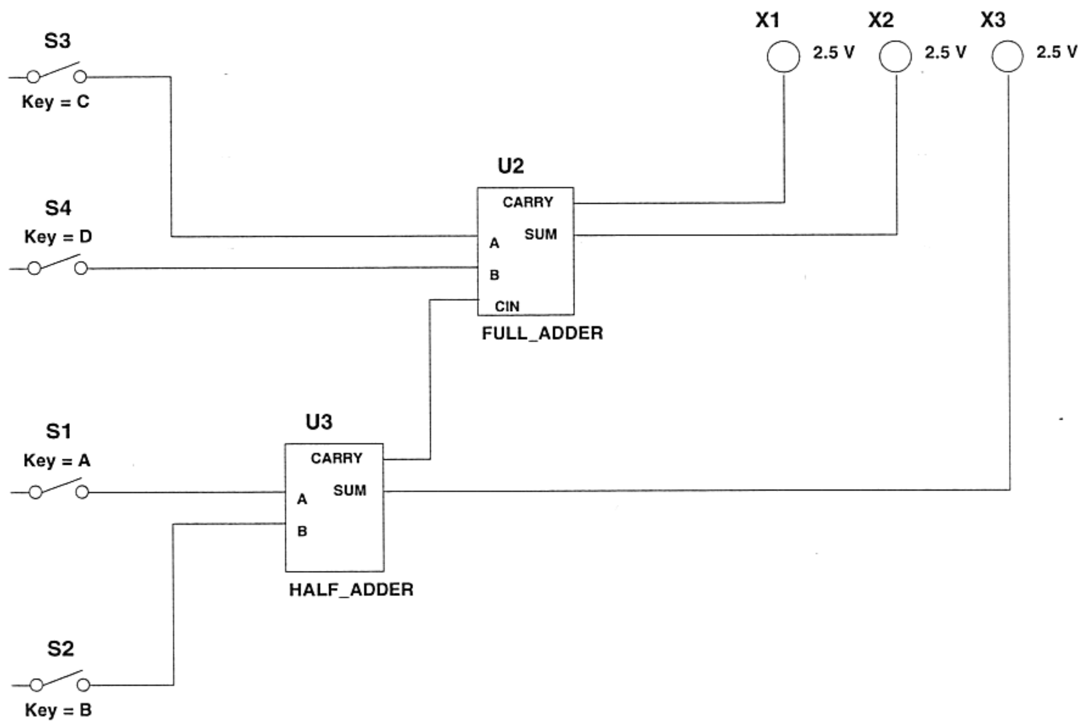


Title: Digital Circuits ECED 2200		
Laboratory #2		
Designed by:	Document N	Revision
Checked by:	Date	Size
Approved by:	Sheet 1 of 5	



Title: **Digital Circuits ECED 2200**		
Laboratory #2		
Designed by:	Document N	Revision
Checked by:	Date	Size
Approved by:	Sheet **2** of **5**	

Chained Half and Full-Adders



Title: **Digital Circuits ECED 2200**		
Laboratory #2		
Designed by:	Document N	Revision
Checked by:	Date	Size
Approved by:	Sheet **5** of **5**	