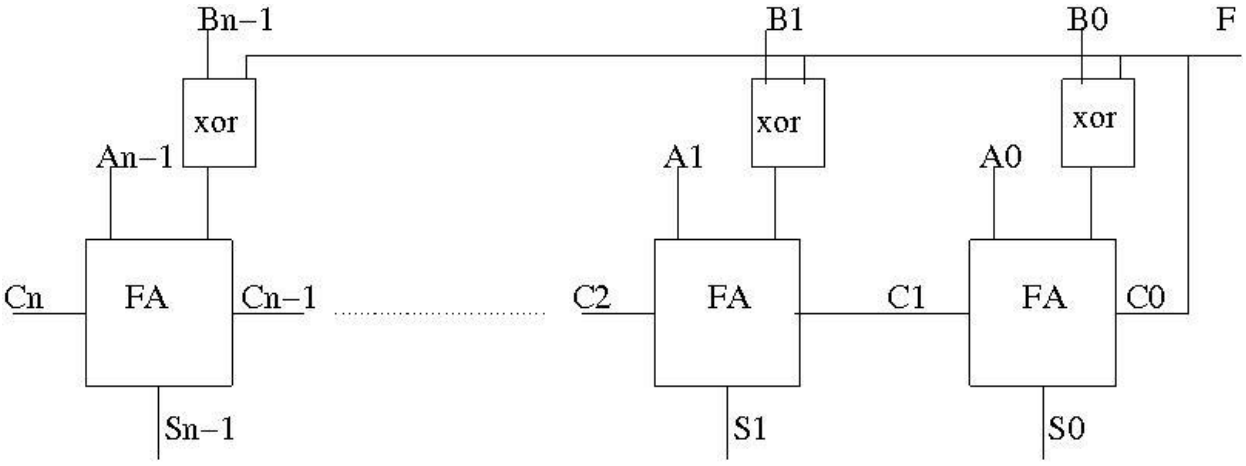
P2

Adding and Subtracting

This lab is going to give you more practice at organising and constructing C++ programs. Its chiefly about constructing functions with clear interfaces and reusing them to build up more complex structures. At the end of this lab you will construct a program that adds and subtracts by modelling a digital circuit, simulating the circuit below with the addition of overflow detection.



Add&subtract with ripple propagation

Schedule

Preparation time : 3 hours

Lab time : 3 hours

Items provided

Tools : None

Components : None

Equipment : None

Software : Eclipse and MinGW

Items to bring

Essentials. A full list is available on the Laboratory website at https://secure.ecs.soton.ac.uk/notes/ellabs/databook/essentials/

Before you come to the lab, it is essential that you read through this document and complete ***all*** of the preparation work in section 2. If possible, prepare for the lab with your usual lab partner. Only preparation which is recorded in your laboratory logbook will contribute towards your mark for this exercise. There is no objection to several students working together on preparation, as long as all understand the results of that work. Before starting your preparation, read through all sections of these notes so that you are fully aware of what you will have to do in the lab.

Academic Integrity – *If you undertake the preparation jointly with other students, it is important that you acknowledge this fact in your logbook. Similarly, you may want to use sources from the internet or books to help answer some of the questions. Again, record any sources in your logbook.*

Revision History

February 3 2013	Jeff Reeve (jsr)	First version of this lab created

1 Aims, Learning Outcomes and Outline

This laboratory exercise aims to:

- Introduce you to the notion of references
- Enable you to think about how you construct complex code
- Enable you to think how digital circuits might be simulated

Having successfully completed the lab, you will be able to:

- Construct complex code from simple components.
- Use the const qualifier.
- Use references in functions.
- Simulate simple circuits

You are shown below how a adder/subtracter is constructed from basic components. The final deliverable of this lab is a program that can simulate the operation of a digital circuit that can add and subtract numbers of arbitrary length. You will need to think about how you interface you program to the outside world with sensible input and output formats.

---

2 Preparation

Read through the course handbook statement on safety and safe working practices, and your copy of the standard operating procedure. Make sure that you understand how to work safely. Read through this document so you are aware of what you will be expected to do in the lab.

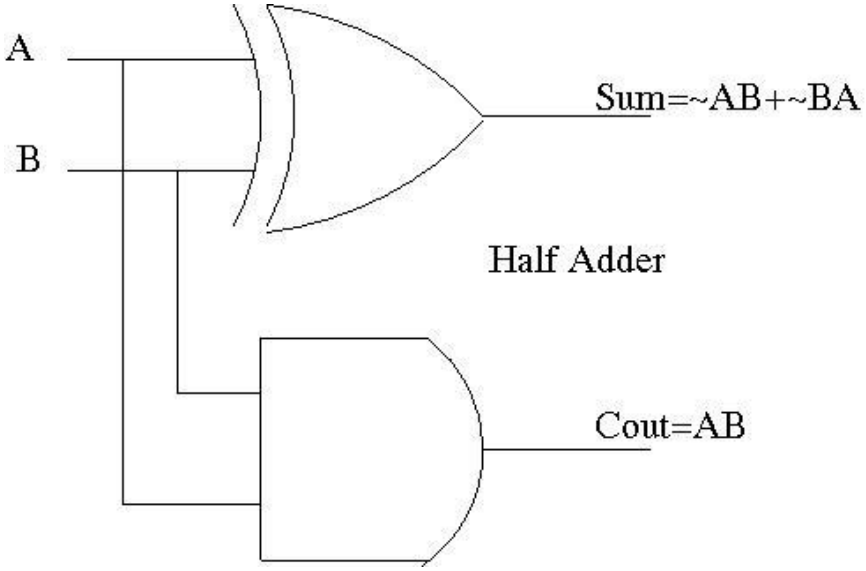
2.1 Preparation Section 1

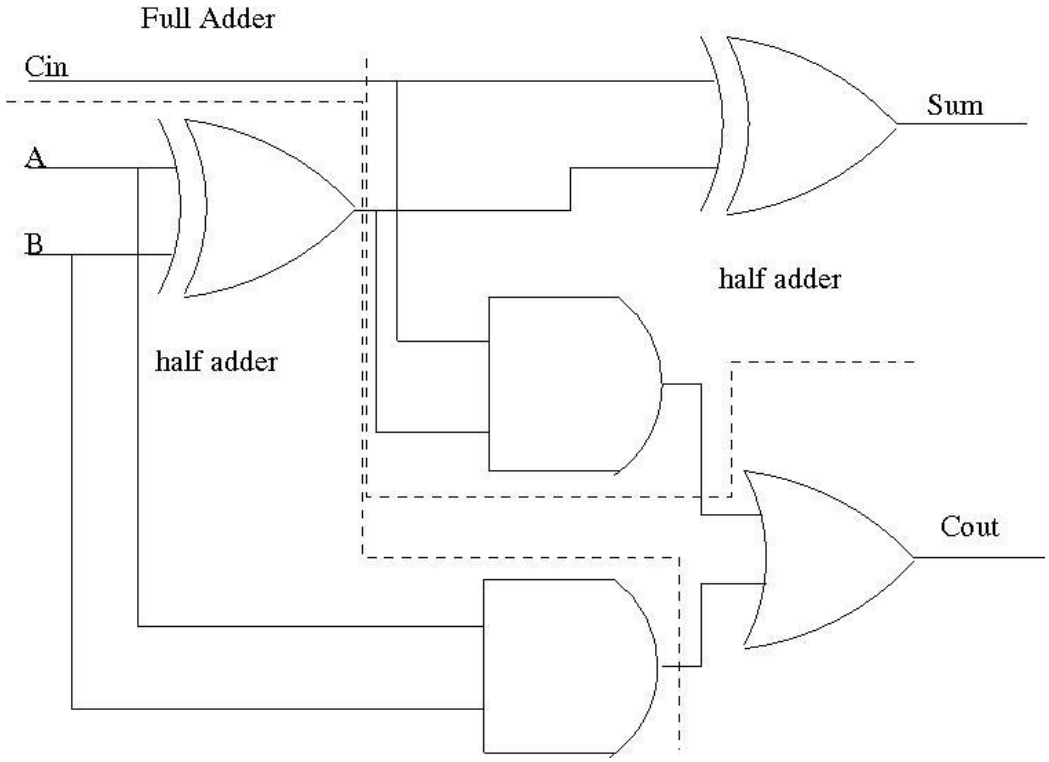
Study the circuits below to appreciate how an adder is constructed and draw figures showing how you will construct the final program. In the diagram below the Sum is the exclusive or of the inputs (~A means not A). Explain how you are going to input and output numbers using streams.

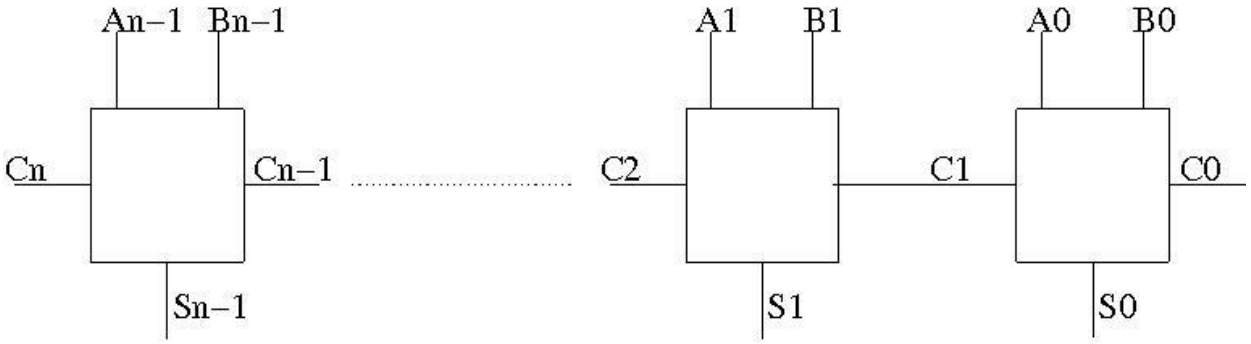
The final circuit is the one on the front cover. When F is 0 the circuit adds (A+B) and when it is 1 it subtracts (A-B).Preparation is essential and a compulsory part of every lab. Overflow is detected when V = Cn-1 XOR Cn is 1. (Not shown in the picture)

2.2 Preparation Section 2

You also need to write test functions to show that your answers are correct. For this show how you are going to convert bit streams to numbers.


A
B
Sum=~AB+~BA
Half Adder
Cout=AB


Full Adder
Cin
A
B
Sum
half adder
half adder
Cout



Multi−bit adder with ripple propogation

---

3 Laboratory Work

Write and test programs that simulate and test all of the circuits above.

---

4 Optional Additional Work

Marks will only be awarded for this section if you have already completed all of Section 3 to an excellent standard and with excellent understanding.

Design and write a program that simulates a multiplier as a cascade of adders.

Appendices

References

HTTLAP

C++ web reference