CS 273 Lab 4: Conditionals and Loops

Dept. of Computer Science, NMSU

Fall 2006

Due Date: Next class after your lab session.

For this lab, you need to program conditionals and loops. The HC11, like all CPUs, has what are called conditional branch instructions. These instructions make the program jump to some other instruction rather than just keep executing down the sequence, but they only do this if some condition is true. The condition is set up by the instruction previous to the branch instruction – a compare instruction. So, it goes like this, for example: If you want to make your program do something if the value in B is larger than the value in A, you can do:

```
CBA
BHI   label
```

The `CBA` instruction means *Compare A to B*. This instruction sets some condition codes, and then the branch instruction `BHI` (*Branch Higher Than*) checks those condition codes, and branches to the instruction on the line that `label` is on if the value in A is larger than the value in B. If the value in A is smaller than (or equal to) the value in B, then the `BHI` instruction does nothing, and the program will continue to execute instructions in sequence (i.e., the instruction right after the `BHI` will execute next). There are many other compare instructions (all begin with C) , and also other branch instructions (all begin with B). We will learn all about how they work, but for now, try using some of them to do this lab.

Another very useful hint for this lab is this: Try first writing down (on paper) the C or Java code that will solve the problems in this lab. Then once you have the programs in these languages correct, convert them to assembly code.

As with previous labs, you should use the `FCB` directive for declaring input values to the program. Give each input value a name (i.e., the label on the `FCB` directive), and then the desired value. To change input values, edit your program and change the `FCB` values, then re-assemble it and run it again. **Don't forget to re-assemble your program every time you change the values!**

Problem 1

Suppose you are programming an HC11 that is in an automated blood cholesterol screening machine. The machine measures the amount of cholesterol in someone's blood and then classifies them into a category. You can visit a web page at

`http://www.americanheart.org`

to see how this is done. We are using a simplified decision set for this lab. There are two types of cholesterol in the blood – good (HDL) and bad (LDL). Cholesterol is measured in mg/dL (milligrams/deciliter). The categorization is based on the total amount of cholesterol and the amount of HDL (good) cholesterol. The categories are:

1. Total cholesterol less than 200 mg/dL and HDL 40 mg/dL or higher. This is the best category to be in.
2. Total cholesterol less than 200 mg/dL and HDL less than 40 mg/dL. Your cholesterol might be too low!
3. Total cholesterol 200 to 239 and HDL 40 mg/dL or higher. Increased risk of heart disease.
4. Total cholesterol 200 to 239 and HDL less than 40 mg/dL. Further increased risk.
5. Total cholesterol 240 and above. Highest risk category for heart disease.

Your task is to write the assembly code which will take two 1-byte unsigned input values, one for the total cholesterol and one for the HDL cholesterol. Your program must have a single variable which will hold the result – your program should set this variable to the number of the category (1-5) that the input classifies the person into. For example, if your program is given the values 159 for total cholesterol and 33 for HDL, then your program should store a 2 into your classification variable. Although total cholesterol can be up to 300 and above, we are only going to use a 1-byte value to represent it, which means the highest test value we can use is 255 for total cholesterol. For this program to work, you must use unsigned conditional branches (`BHI`, `BHS`, `BLO`, `BLS`). If you use the signed conditional branches (`BGT`, `BGE`, `BLT`, `BLE`), your program will not work.

Questions:

NOTE: For questions 1-5 you will have to change your program to use the different values, reassemble it, and then run it with the simulator. When submitting the program, only submit the one with the input values for question 5. Also, you will be asked about the time counts for each run, so it would be a good idea to write them down.

1. For test values of TOT=187 and HDL=17, what is the final value of your result variable (in hex)?

2. For test values of TOT=163 and HDL=52, what is the final value of your result variable (in hex)?
3. For test values of TOT=200 and HDL=44, what is the final value of your result variable (in hex)?
4. For test values of TOT=251 and HDL=46, what is the final value of your result variable (in hex)?
5. For test values of TOT=239 and HDL=40, what is the final value of your result variable (in hex)?
6. What is the final time count for your program when run with the inputs for question 3?
7. (a) Which of questions 1-5 produced the longest running execution?

   (b) What was the time count for that?

Problem 2

For this problem, you must calculate the sum of the integers from a given lower bound to a given upper bound, inclusive. Your program will take two signed values, and you will need to add up all the integers between and including those two values. For example, if you are given -3 and 6, then you should add up

$$(-3) + (-2) + (-1) + 0 + 1 + 2 + 3 + 4 + 5 + 6$$

and produce the result of 15. You do not need to come up with any clever solutions, you can just do a straightforward loop. You can assume that the first value given is always less than the second (i.e., you don't need to figure out which one is less). Also, you can assume that the resulting sum will fit into a signed 1-byte integer.

Questions:

1. For 7 and 14, what is the final result in memory (in hex and in decimal), and what is the time count (in decimal)?
2. For -12 and 3, what is the final result in memory (in hex and in decimal), and what is the time count (in decimal)?

Lab Turn-in

For this lab, you must hand in the following (as separate files):

1. source code file for problem 1 with the input values for question 5,
2. source code file for problem 2,
3. lab report written in the required format, including
   (a) answers to the questions in problem 1.
   (b) answers to the questions in problem 2.

Submit these through the Web Submission Page at

`http://www.cs.nmsu.edu/~joemsong/273/upload/`

Due Date: Next class after your lab session.