

Homework 3
UCR EE/CS120B: Introduction to Embedded Systems
Fall Quarter 2004, Lecturer Brian Grattan

Due Tuesday, Oct. 26th at the 10PM via WWW turnin.

Name: _____

UCR ID#: _____

For this assignment, you can, if you choose to, work in teams of **two** as long as your partner is not your lab partner (we're trying to get you to branch out). If you do work in a team, indicate who you worked with in the comments in your code.

1. (20 points) Create assembly code for the 8051 that calculates the same thing that we did as a single purpose processor in HW2:

$$area = \sum_{i=1,1+a,1+2a,1+3a\dots}^d i^2 a$$

This time, your inputs 'd' and 'a' will be registers R3 and R4, and your output will be R5. In your code you can just set R3 and R4 at the beginning of your code. Comment every line! Have some comment on each line that indicates what you are doing.

You will create the assembly, convert it into a .hex file, and then simulate it on the simulator to verify that it works.

The steps for creating and simulating your code are as follows:

- 1) Log into any lab machine in the Computer Science Department and start Windows (type "Windows" from the command line inside a terminal if you're on a Linux machine). Alternatively, you can work from home by using Windows Remote Desktop to access "windows.cs.ucr.edu". Google for "windows remote desktop" if you're confused.
- 2) Now we need to access Keil 8051 tools that are found on a machine called peart. From Windows Explorer go to "tools" / "map network drive". Choose drive P:, and in the folder box type "\\peart.cs.ucr.edu\8051", then click ok. You should now be able to access the 8051 tools in P:\.
- 3) From the Start menu choose "run" and enter "cmd" in the box and press enter. This will bring up a command prompt. Now type "P:\cs122.bat" at the command prompt. This will setup the compiler environment variables.

- 4) You can write your assembly code in plain text using notepad or your favorite editor. Just remember to put each instruction on a new line, and use semicolons for comments. For the first time around use the provided 'example.asm'
- 5) To assemble your code, you will need to run a51 from the command line. Type "a51 example.asm" from the command line. If there are no errors then this will produce example.obj.
- 6) Now we need to link our code and produce a .hex binary file. Type "bl51 example.obj" from the command line. This will produce a file called "example" or maybe "example.omf". Finally, run "oh51 example" or "oh51 example.omf" depending on what file it created to produce example.hex. Now you've created an 8051 hex file!! But does it work correctly?
- 7) To test your 8051 hex, you'll need to download the 8051 simulator here: <http://bit.kuas.edu.tw/~8051/emul8051.zip> and install it. Testing your code should be quite easy if you've ever used a debugger before. To load your hex go to 'file' -> 'load hex file', and navigate to the hex file you created (example.hex). Use Step(F11) to execute your instructions one at a time, and watch the effect of the instructions on your registers. Does your code perform as expected? If not, try to figure out what's wrong, fix it, and simulate again.

HINT!!! – If you get stuck, it may be helpful to use the Keil c51 C compiler to produce 8051 assembly from C code. To compile a C source file type "c51 file.c src" from the command line. This will produce file.src with the assembly code. If you look at file.src you'll see that it also contains some other information at the beginning and end of the file – if you ever want to assemble and link this file, you'll have to delete this extra stuff so that you have a file of assembly instructions only. Basically, delete everything above "main:" otherwise you'll get assembler errors!! And no, you can't just write the code for the algorithm in C, run it through C51, and turn it in. However, doing this may not be a bad starting point!!

Some questions that may come up:

- Do we have to worry about special cases ($a > d$, area gets too large and overflows, a or d is negative, etc.)? Answer: No, but your code should work for any combination of 'a' and 'd' where $a < d$ and the answer does not overflow.
- Are 'a' and 'd' always going to be positive integers? Answer: Yes.
- Isn't it pointless to do integration when your accuracy can never be less than one? Answer: Stop asking so many questions and just do the assignment.

Useful Links

- Keil 8051 support page - <http://www.keil.com/support/default.asp>.
- 8051 Instruction Set Reference - <http://www.win.tue.nl/~aeb/comp/8051/set8051.html>
- Google – <http://www.google.com>

What to turn in:

Your assembly code (in text format) and your .hex file created from that assembly code for the inputs a=1 and d=5 (your code should work for more than just this case though). If you work in a team, each person should turn it in using their login.

Here is a screen capture of the simulator in action:

