07 Oct. 2015

Chapt. 1 CENG 255 TA Lab Log Progress Report

TA: Philip B. Alipour

General Log and Notes:

1. What happened in the lab?

i. Last week session started at 12:00PM. The formal duration is 12:15 to 3:15PM according to http://www.ece.uvic.ca/~ceng255/lab/information.html#schedule.

ii. Total of 23 students were present, where pre-labs will be marked for tomorrow's session. The 2nd experiment session will be conducted with all students marked prior to report submission in two weeks. Next week there will be no labs while the room is accessible by a keycard.

iii. Marking results handouts and comments based on our agreement will be given to each individual with only their student number visible on screen or posted on the current webpage with the individual's grade.

iv. Students should follow every single step properly from the lab manual, however, I and **Brent Sirna** (the technician) will point out about any updates in the upcoming lab manuals. So I do apologies if some students have to conduct the experiment under a bit of stress or hurry, but eclipse debugger environmental setting requires a bit of patience.

v. The major issue is to get the debugger configured properly according to Lab #0 steps in its manual. I continue to assist students on their machines as problems occur. Of course, **Brent Sirna** was around and will be there for rectifying extreme technical issues if they occur.

vi. During the 2nd Lab session I will be focused on students' **lab performance** to make sure the material and exercises are done accordingly.

vii. I will include bonus points, a maximum of 5% in order to compensate the overall session grade for those who have lost points during and after session (such as in the lab report due in two weeks.)

viii. Students were informed about the pitfalls, and hinted about the solution whilst the program transformation process, memory structure, debugging and objectives of the lab were explained for each group through interactive QAs exchanged between the TA and the student.

Notes for the Attending Students on the forthcoming lab sessions:

1.1. Notes to consider on the next lab: Lab 1

Lab manual and the relevant files are available at http://www.ece.uvic.ca/~ceng255/lab/.
Make sure to download all including the metadata and save into your **workspace**. Your workspace is created once you run (execute) Eclipse under your station username. However, based on my experience, the communication between the **microcontroller** and the **workstation**/PC sometimes created unstable responses, hangs and thus an on-hand real-time debugging solution is required. On occasion (especially when Leonard had his group session), I/student had to disconnect the microcontroller from the station and resume all over again by logging off and take the same steps.

To investigate memory contents thoroughly between the memory map in eclipse and thereby study the hardware firsthand, run the **STM-32 hardware program** on your desktop. You may depict and mention these comparisons in your report if it helps your discussion/analysis on the lab parts.

1.2. Hardware/software challenges

- Make sure to **terminate** your debugging algorithm when you aim on running another lab. The program runs in the following order according to Lab #0 manual steps in your Debuggers Config. settings: ***Monitor reset halt***. In other words, multiple labs/programs cannot load concurrently on the microcontroller when one is still initiated/engaged and not terminated. The algorithm terminator is indicated with a **red square** at the output stage or delivery stage of the program (refer to the bottom flow diagram of your lab manual where you should explain the logic behind the linker, program transformations, etc. in your report).
- In the diagram from Lab #1, you may include where and when the **linker** is invoked (at the end, throughout or wherever… explain) as well as the libraries. **Hint:** refer to your lecture notes on the Assembly Introduction.
- More about the linker and to ascertain how eclipse works with the compiler and linkers, commands and options/switches visit: http://tigcc.ticalc.org/doc/gnuasm.html
  more information is on the same lab webpage
  http://www.ece.uvic.ca/~ceng255/lab/

There are two ways to **link**:

Automatically via **Eclipse**:
1-Go to project explorer
2-Build project or build all under project build configurations.

Manually via **command-line** (http://en.wikipedia.org/wiki/Command-line interface and http://tigcc.ticalc.org/doc/comopts.html) in Windows OS using the **gcc compiler**, one can type e.g.: If we want to link files to an executable we conventionally type:

```
gcc -o  output  file1.c  file2.c
```

The manual approach will make you certain that you have learned how the transformation from one program/machine level to another is done (like the Debug. Config experience).

- The **i->** button must be ON if you want to view your entire code (high level source code + assembly) in the disassembly window (perspective).
- Put the exact memory address in the memory windows you create and list: e.g. don't forget to include the standard prefix **0x**… for a targeted memory address.
- If the program is generating **errors** after building your project at the debugging stage, make sure the debug configuration is properly set according to Lab # 0 manual. If needed, **sanitize** everything, by cleaning the entire workspace: totally remove all of the files, and reinstall (save) the lab folders as you close Eclipse and restart the process from the very beginning. Patience is a virtue!
- The **nop** command must be typed in between breakpoints if you are using step into line, step over line of code commands at the output stage (as you perceive all of the perspectives on display).

2. The lab report + additional session experience for a good lab performance

- The report must abide by the content structure required in your lab manual according to http://www.ece.uvic.ca/~ceng255/lab/report.html#contents
- I suggest, you take a screenshot of the overall view of the output which includes **program listing** (from the **disassembly** view where all parts/components of the source code is decomposed and available. See figure 3 of the lab manual), **memory map**, register values etc.
- If you have changed the code, or have put new commands or have made code modifications, **terminate** the current program and do these changes in the **C/C++ perspective**. However, you can also investigate changes instantaneously by changing memory values as you compare different addresses using pointers (refer to your Lab #1 manual, Fig. 5 and Step 5).
- To explain your codes/algorithm in the report, it is better to have them commented (see Lab part 1 as the comments are written in terms of **//**…). Quantify and measure/calculate between addresses in terms of values. Which value would be interesting and why? Are their unexpected values output as function errors when an operator is changed? How would you address it? (Investigate your code relative to **memory content** like a **detective**)
- To save your source file, either copy-paste it in form of a text file and keep it in a safe drive (there is no guarantee whether these workstations will keep your changes/files as the drivers get cloned or updated from the maintenance team/administrators), or attach and email the file to yourself and/or your lab partner or save it on your flash drive. It is your responsibility to avoid any data loss and keep records on your progress.
- For step 6 exercise, refer to Fig. 4 and 6, use **properties** (right mouse button click on the address/value) to see the **little-endian** and other aspects of the value of interest in your memory. Also, change of commands to e.g. **SUBTRACT** should be investigated by converting and comparing values between addresses/pointers and pre-post stepwise executions of the main code (source). This is to see whether the memory content actually changes in that particular address. What values are being changed and would the output affect the program if you change it, or is it done from the source code once a command is changed? These questions should be answered throughout the lab session as you practice and reflect on your report. It is encouraged to calculate manually and compare your result to what is generated by a **calculator**: http://www.mathsisfun.com/binary-decimal-hexadecimal-converter.html

Scalar view of the memory can be achieved by specifying the dimensions, e.g.

```
(int *)&x
```

in your **memory browser** which could be found on your menu, under **window** at the output stage.

Specific scale of the address or the pinpointed version only requires the prefix 0x<No.> as explained above and exemplified in your lab manual.

- Include the memory map (where the range and partitions of the memory are shown) as you examine and investigate your memory conversions from e.g. **binary ↔ Hexadecimal ↔ ASCII**. In that, two or more screenshots

you can have these maps and all perspectives as you refer to the imported figure(s) in your report and explain the process.

Basically, explain how you have achieved your objectives in this lab, such as your experience with the program, TA and maybe other groups. **Note that**, you are allowed to share your experience or approach with others but not the solutions. The solution must be done by yourself or the help of your lab partner and guidance from the TA in achieving your goal.

- In addition, answering to the four questions for this session you can gain **bonus marks** for an extra 5% of the 100% lab mark, which is considered for those who perform well (productive) and knowledgably.

2.1. Deadline for submission: within two weeks from our last lab session, in form of a hardcopy, drop it in my drop box on the 2nd floor in the lobby corridor where my name and our group number are labeled together.

2.2. The marking criteria
I have sub-sectionalized the marking as instructed according to the lab grading webpage: http://www.ece.uvic.ca/~ceng255/lab/grading.html

- For the pre-lab involving **verbal and written Questions, 7.5%** for the first lab part as well as some or all of the main three questions. **7.5%** asking how all or the last two parts of the lab work. This is to make sure you have a fair idea of the functionality of the basics of eclipse, and the algorithm(s) involved.
- **Lab performance is 60%**. For this particular lab, I break it down as follows:
  - **Goal achievement** on part I: 20%, part II: 10% and part III: 15%.
  - **Questions on all parts:** 15%.
- **Report is 25%** which consists of: Detailed program listings 5%; Results (snapshots, memory contents etc.) 15%; Discussion & Conclusions 5%.

and the report structure as well as the deliverables are specified at http://www.ece.uvic.ca/~ceng255/lab/report.html#contents

At the end of the day, it is a learning curve and experience for all of us. I hope we continue to be productive and conduct our sessions with a positive outcome without getting lost from the overwhelming contexts of the course material, hypothetically speaking :) Oh, one last thing, always keep the von-Neumann architecture as your guide to all scalable modern architectures ;)

Have a productive week,

See you tomorrow at 12:15 or earlier,

Philip B. Alipour,
Ph.D. Candidate in Electrical, Computer Engineering and Quantum Physics,
Dept. of Electrical and Computer Engineering, University of Victoria, V8W 3P6, Canada,
Office: ELW Room # A358,
Emails: phibal12@uvic.ca or philipbaback_orbsix@msn.com
Homepage: http://web.uvic.ca/~phibal12/