**ECE 371/372 Project Documentation Guidelines**

**[ I ] Overall organization of documentation (10 points)**

There should be a lot of emphasis on the organization of your documentation. A well-organized report will receive full organization points. A well-organized report is easy to follow with clear indication of what each section shows. The related sections should be in order and not scattered throughout the documentation. In summary, your report should be such that you would like to get it from your co-worker in a company. Before submitting your work, you should analyze your report and think if you would like to read and/or grade the report if it were given to by someone else.

**[ II ] Design Log (20 points)**

The design log ***should be as-you-go and not after the fact***. It should contain the initial research and thinking you did before developing the algorithm. For example: calculating/finding some address, specific control words that correspond to different actions, calculation of frequency/values etc. as you go along, It should describe not only what you did but what were the ***problems encountered, how you solved them*** and the steps involved in solving them. The design log should be detailed but to the point. One should be able follow the development process and **thinking** you did for the project. The organization guidelines from section [I] also apply to design log. Initial algorithm(s) should be included in the design log as you develop them and also a detailed initialization list.

**[ III ] Algorithms (20 points)**

* The idea behind algorithm(s) is that a person with an understanding of ARM ISA and assembly programming should be easily able to reproduce the program following your algorithm(s).
* You will likely have two or more versions of algorithms: initial algorithm(s) and final algorithm(s). The initial algorithm(s) include your first thoughts of the flow that you would use to solve the problem. This may be incorrect and this is fine. ***The final algorithm(s) are the correct algorithm(s) and correspond to the correctly working program that you demonstrated to the TA****.* If initial algorithm was incorrtect and you need to debug it, the debug steps and thinking between the initial algorithm and final algorithm should be part of your design log (as mentioned in section [II] ).
* It is very important that your final algorithm is ***generic and at an abstract level***. The initial algorithm does not need to follow these directions. Although the final algorithm must be abstract/higher level, it should show all the control word and addresses needed to perform an operation, however, it should not detail the specific general purpose registers you used to carry out that particular operation. For example: Say, your program has some interrupt service routine and the first thing you do in it is to clear the cause of interrupt i.e. the edge detect. ***The following two ways of writing it are unacceptable***:
  + Clear the button interrupt edge detect *or*
  + Load address 0x123456 into R1, read from 0x123456 into R2, AND R2 with mask 0x0010000 and write the result back to 0x123456.

The first way does not provide you the complete information and second gives you the steps to perform the operation but does not tell you what are you trying to accomplish. An acceptable example to write this is: ***Clear the GPIO 91 button interrupt edge detect by writing 0 in bit 17 in GEDR3 at address 0x40E00148*** *.*  This provides a reader complete information in an abstract manner and reader can use his own steps to clear bit 17 (either by ANDing or using BIC, thus your specific mask are not necessarily needed).

* Since the algorithm is written at an abstract level, any use of general purpose registers in describing the algorithm would result in points taken off. For example: Load something into R1 and decrement R1 till 0 is not acceptable. The specific registers and memory locations are absolutely okay to be mentioned like GEDRx, GRERx etc.
* The final initialization list must be shown right before the final algorithm or can be included in the beginning of algorithm as a part of it.

**[ IV ] Working Program (50 points total, 40 working program, 10 comments)**

It includes three parts: 1. Successful demonstration of the program meeting all the requirements, 2. Correct answers to the question during the project demo and 3. Clear comments in the code.

Part 1 is self explanatory. For part 2, a student should be able to answer some simple questions related to the project being demonstrated to show that he/she not only got the program to work but also understood the material. For part 3, students should make sure that the comments are correct in their final version of code. Since students initially use the comments from the book, once they modify the program for their project, they often forget to update the comments of the sections updated. Also, the comments should correspond to the final algorithm for example: if algorithm says to clear bit 31 but program comments say bit 30 then some points will be taken off for incorrect comments.

**Suggestion:** We have limited number of setup/boards and larger number of students, everyone should get equal time to test their program on a board. If you have been testing on a board for a while with not much progress and another student is waiting to use the board, please give other student some time to test their program. In such a case, the TA can ask you to move. You should try to carefully analyze your program for errors instead of just trying to get it work by hacking away on the board.

***These are general guidelines for the project documentation and you should make your own reasonable decisions using the “golden Rule of documentation”, for anything not covered here.***