General Lab Policies and Procedures

ECE341, University of Idaho

Spring 2015

Lab Preparation

- You are expected to come to your assigned lab section prepared to start writing code. This means that you have read and familiarized yourself with:
  - The lab handout.
  - The datasheet and/or reference manual for each new peripheral, protocol, component, and software libraries we are working with.
  - The relevant chapters in the course textbook.

- You are strongly encouraged to start the labs early as they can be deceptively difficult.

- These labs build on each other. If you don’t get this week’s lab working then next week’s won’t work either.

Deliverables

For every lab, there are three parts. Each part is due at a particular time, and worth a certain portion of the total lab grade. Late work is not accepted; the electronic filters will discard everything received after the deadline. An example of a good report may be provided.

<table>
<thead>
<tr>
<th>Submission</th>
<th>Deadline</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prelab</td>
<td>Tuesday, 8:00 AM</td>
<td>20%</td>
</tr>
<tr>
<td>Demonstration</td>
<td>Friday, 3:00 PM</td>
<td>10%</td>
</tr>
<tr>
<td>Report</td>
<td>Monday after the Demonstration, 5:00 PM</td>
<td>70%</td>
</tr>
</tbody>
</table>

General expectations are described in the next three sections about the prelab, the demonstration, and the report. These are the expectations for every lab deliverable, unless otherwise stated on the webpage, in class, or in lab. Each lab will have a lab handout that
explaining concepts related to that week’s lab; these handouts will also contain specifications that must be met for a satisfactory demonstration. When necessary, there will be an additional document posted along with the handout that will help clarify the handout and may modify the expectations below. This additional document will also contain topics and questions that should be addressed in the report.

Prelab

- The prelab is a planning tool to prepare you for the lab. It will consist of two parts: the first is the goal and background information, and the second is your plan to accomplish the goal.

- The goal must be consistent with the lab specifications and should be short, clear and concise.

- After the goal, the prelab should elaborate and expand upon subsystems mentioned in the goal, providing background information.

- The background information should be a summary (not copied) of the relevant prelab materials (lab handout, datasheets, textbook, and lecture).

- You must cite your sources in the background summary. It doesn’t matter how you cite it (footnotes are fine), but it must be clear where the information comes from.

- Do not simply regurgitate (or worse, copy) the lab handout!

- Only write background information for new components or peripherals.

- The first part of the prelab should be copied into the report as the beginning of the introduction (after it has been updated and corrected if necessary).

- The second part, your plan, will form the groundwork for the new material added to the implementation section of the report.

- The plan details how you intend to accomplish the lab and meet specification.

- While working on the lab, it will be helpful to go back and modify your plan to document your code. This will make the inclusion of the new implementation details into the report trivial. However, don’t let modifying your plan take priority over completing the lab and demonstrating it.

- The plan must contain at least two distinct diagrams: one that defines how data flows through the system (the data flow diagram), and another that defines the execution flow of the program (the control flow diagram). Additional information about the diagrams will be provided if necessary.
• The prelab grade will depend on how well the background material is understood, how mature the plan is, and the quality of the diagrams.

• Unless otherwise stated, the prelab shouldn’t be more than a page and a half, excluding diagrams. While there is no minimum length, make sure it is long enough to satisfy the requirements above.

Demonstration

• You are responsible for proving your code meets specification.

• You must come up with a set of tests that you’ll show the TA. Each test will show at least one specification being met.

• The tests you use to demonstrate your lab to the TA should be subset of the tests you describe in the report (see below).

• During the lab, the TA will be unable to thoroughly test your design and ensure complete functionality. Because of this, as soon as you have gone through your tests and the TA is satisfied, you will submit the project via email.

• The project you submit must be the one you demonstrated and the one described in the report.

• To submit the project you will first rename the project (if necessary) to the appropriate name, then package the project (by right-clicking on the project name in the Projects pane, typically on the left) and email the resulting .zip file to the TA. Refer to the first appendix (Appendix 1) of this document for the step-by-step procedure to correctly submit the project. Refer to the section named “Submission Formats” for the correct subject line and filename formats.

• All .c and .h files must be in the compressed project folder for your lab to compile after it has been decompressed.

• When adding files to your project, check the “Copy” checkbox when you’re browsing for the specific file. This copies all added files to the working directory (project folder) and prevents a whole family of linking and build errors as well as making it possible to correctly submit the project.

• The project folder should also contain other files in addition to your .c and .h files. The files are listed below, but you do not create them, they are created by MPLAB X automatically when you make a new project. A complete directory and file structure is not provided because it is dependent upon versions, how you name your files, the project history (compile times, object file names, etc) and other options in the
configuration. However you must always make sure you have a file named Makefile or something very close, and a folder named either nbproject or .nbprojects.

- You are responsible for making sure the zip archive can be downloaded, unzipped in any directory, opened in MPLAB X and compiled without error.
- Partial credit will be awarded for demonstrations that don’t meet specification, just make sure to note the parts that don’t work both when you demonstrate and in the report.
- The only way to get a zero for the demonstration is if you don’t demonstrate or submit your demonstration on time.

**Report**

- The report should be a formal, professional report that is clear and concise. It is not required to typeset the report, but reports typeset with \LaTeX will be awarded extra credit.
- The report will consist of four major parts: the introduction, implementation discussion, testing and validation, and the conclusion:
  - **Introduction:**
    * Will be primarily copied from your prelab, as long as you’ve kept your prelab plan updated and corrected any misunderstandings in the background information and the goal.
    * Should only contain background information about new peripherals or components.
    * Should transition smoothly into the implementation.
  - **Implementation**
    * The implementation will document and justify the code you have submitted for your demonstration.
    * You will integrate new modules that are introduced each week into the report as well as update old modules if necessary.
    * Must include both a data flow diagram as well as a control flow diagram that describes your code.
    * The diagrams will become complex; use hierarchy to keep them readable.
    * Must include the majority of your code from the entire lab in the form of “listings” (similar to the lab handouts). The listings generally shouldn’t be longer than a half page, should be captioned, and discussed in the text surrounding it.
* Must describe the role each listing has in the system (its purpose).
* While this section will become quite large, each week you will only need to add documentation about the new code you wrote for that lab each week.

---

Testing and Verification

* You are responsible for testing your own project.
* You must come up with tests that can show if your code meets the specifications of the lab.
* You should describe your tests in such a way that they are easily repeatable (by the TA).
* For each test, you should perform that test on your design, document the result, and explain the implications of the results.
* Remember the instrumentation you have and use the appropriate type for the test. For example, don’t use a LED to determine if a delay is exactly one millisecond.
* The oscilloscope is a very powerful tool and screen captures are great for showing signal timing. After Lab 1, new screen captures from the oscilloscope should be included with every lab report (unless stated otherwise).
* Each test should prove or disprove at least one specification. Examples will be discussed in lab if there are questions.
* These tests should be developed while writing code for the lab. For example, if you write a function that writes a character to an LCD, you should test that function to determine if it works before using that function in another that writes a string.

---

Conclusion

* This will vary the most from lab to lab, but it will always contain a discussion about the limitations of your design and the microcontroller. For example, if we are working with a timer, what is the maximum interval we can measure? What is the minimum interval we can measure? Always take parameters to the limit and ask, “What is the limiting factor here and why?”
* Must address specific topics and questions specific to the lab which will be posted with the lab handout.
* The answers should flow together in a few paragraphs with full sentences. Sentence fragments will not be accepted.

Submission Formats

* All submissions (the prelab, demonstration, and the report) will be via email; the email address is ece341submissions@gmail.com
• Every submission will have a subject line that begins with `[ECE341] [LAB#]` with the # symbol replaced with the actual lab number.

• The third part of the subject line it will either be `[PRELAB],[PROJECT]` or `[REPORT]`, depending on the type of deliverable you are submitting.

• Your last name then first name will immediately follow the type specifier and be separated with an underscore: `Lastname_Firstname`.

• For the report and prelab, the filename of the attachment should be a copy of the subject line with `.pdf` appended. For the project submission, the attached file should be named similar to the subject line, except the [PROJECT] part is omitted and it must end with `.zip`.

• There should only ever be one file per submission email.

Here are a couple of examples:

Kyle Swenson’s prelab submission for Lab 3 would have the following subject line and attachment:

Subject: `[ECE341] [LAB3] [PRELAB]Swenson_Kyle`  
Attachment: `[ECE341] [LAB3] [PRELAB]Swenson_Kyle.pdf`

James Nelson’s report submission for Lab 11 would have the following subject line and attachment:

Subject: `[ECE341] [LAB11] [REPORT]Nelson_James`  
Attachment: `[ECE341] [LAB11] [REPORT]Nelson_James.pdf`

Ronald McDonald’s project submission for Lab 8 would have the following subject line and attachment:

Subject: `[ECE341] [LAB8] [PROJECT]McDonald_Ronald`  
Attachment: `[ECE341] [LAB8] McDonald_Ronald.zip`

Attendance

• Attendance is mandatory.

• Every minute you are late you will be penalized 5% of that lab’s entire grade. This means that 20 minutes late will carry a penalty of 100%.

• If you satisfactorily demonstrate your lab to the TA before the end of the lab session, you may leave the lab session early.
• Important and helpful information will typically be presented at the beginning of the lab session- information such as what is required in the report, clarification on specifications for the lab, as well as the questions and topics you should address in the report conclusion.

• The lab section is your time to work on the lab with the TA available- take advantage of this fact!

Plagiarism

• The labs are not a team-based effort, they are individual labs and you are expected to work alone and come up with your own designs.

• If you copy code/reports/diagrams/prelabs and it is the first offense, you will receive a zero for that particular lab.

• If you copy code/reports/diagrams/prelabs again, you will receive a failing final lab grade.

• If you are ever in doubt about what constitutes plagiarism, ask the TA. If you ask the TA and it turns out it is plagiarism, you will not be punished as long as you don't submit the code. Plagiarism is only punishable if it is submitted for grading.

• You are allowed to
  – Look at code in the book, in the datasheets, in handouts, and from the lecture and use that code as a guide for you to write your own code. If you do, informally cite the source in a comment.
  – Refer to code on the web only if it is for referencing the syntax or semantics of the C programming language (how to write a switch-case block, for example) and standard libraries (the function printf, for example).
  – Help your classmate with general semantics of the C programming language. For example helping a friend with pointers is just fine; telling a friend how to correctly initialize the I2C bus is not.
  – Help each other by fixing syntax errors

• You are NOT allowed to
  – Share code
  – Copy code from the internet
  – Look at code pertaining to the lab from the internet. There are past labs that are available online; if you are caught looking at, referring to, or otherwise using those labs, it will be considered cheating and these rules will apply.
– Copy code from the book
– Write code you don’t understand
– Help your classmates with lab-specific details

• You must understand all the code you write for these labs.

• If some code looks suspect, you will be confronted and asked about it. A discussion will take place that will essentially assess how well you understand the code in question. If you don’t understand the code you turned in, it will likely be considered plagiarism.
Appendix 1 : Submitting your Project

This appendix describes the steps necessary to successfully submit your project. It is a good idea to make a backup copy of your lab before you rename and package it just in case something goes awry.

1. After you have demonstrated your lab to the TA, right-click on the project name in the Projects tab (typically on the left) and select Rename... (Figure 1a).

   ![Step 1.0 - Rename the Project](image1)

   (a) Step 1.0 - Rename the Project.

   Figure 1: Renaming the project

2. Rename your project to the correct format (shown in Figure 1b). Your project name should be renamed to the following: [ECE341][LAB#]Lastname_Firstname (with the # symbol replaced by the lab number, and the Lastname_Firstname replaced with your last name and first name).

3. After the project is renamed, right click the project name again and select Package (Figure 2).

![Step 1.1 - Rename Project Dialog](image2)

(b) Step 1.1 - Rename Project Dialog
4. Packaging the project will create a compressed file with all of the necessary source code for your project. This .zip file can be found in your project’s directory and will serve as your project submission. Make sure the email’s subject line satisfies the requirements specified in the “Submission Formats” section of this document. For your convenience, the subject line should be: [ECE341][LAB#][PROJECT]Lastname_Firstname and the attachment’s filename should be [ECE341][LAB#]Lastname_Firstname.zip.