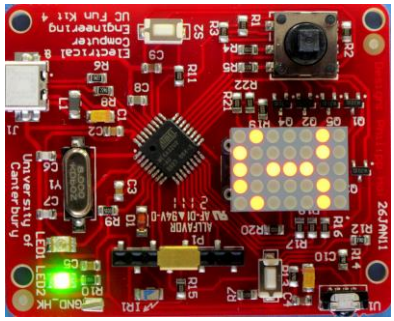
ENCE260 Embedded Programming Assignment

September 2017



Due: Week-ending Friday 20th October 2017
Worth: 10% of final grade
Groups: Students work in assigned groups of two.
Purpose: To learn about simple, modular C programming on a small-scale embedded system.

1. Introduction

You are to use the API functions to write a small set of programs that will implement a simple multi-user game[1] for your UCFK4 microcontroller boards. The goals of the assignment are:

1. To program a small-scale embedded system in a modular fashion.
2. To use the modules provided to abstract hardware dependence.
3. To implement a simple real-time infrared communications protocol.
4. To implement at least one C module, each.
5. To follow the ENCE260 coding style standard.

2. General instructions

Here's what you need to do:

1. You will be allocated a project partner by Learn's booking system in Week 2. It is your responsibility to check the details of your lab partner and contact him or her as soon as possible. Your group is registered so each group will have an individual group number. Your git repository name is based on your group number (see Sect. 5.5). Login to your git repository and start using it from Week 3. Your code cannot be marked until you have a git repository and have committed your code to your respective group repository. Contact Richard Clare (richard.clare@canterbury.ac.nz) & Patricia Inez de Andrade (patriciainez.deandrade@canterbury.ac.nz) if you haven't been allocated a partner by Week 3 or cannot contact your partner.
2. Develop a multi-user interactive game for the UCFK4 using the navswitch, display, and IR communications.
3. Demonstrate your program during your lab during the last week of Term 4. To be fair to other groups you may only demonstrate your assignment to a TA or your Lecturer *during the second hour of **your** respective lab session*.
4. Perform a final commit of your program source code to your git repository no later than 17:00 Friday 20 October for marking.

3. Program submission

1. NB, your game must consist of a single application. If you require different functionality on each board, then this must be determined at run-time, say by pushing a button.
2. Ensure each of your source modules have both group members' names and usercodes in a banner at the top of each source file. *Your code will be assessed in terms of documentation*.
3. Ensure you have followed the required programming style for this project.
4. Ensure you have not changed any of the modules staff have provided.
5. Place all your final application modules in your git cloned directory: /ence260-ucfk4/assignment/teamXX, where XX is your group number, e.g., team132. By doing this you will be able to *periodically* push your source files directly from your teamXX directory to your

[1] Multiuser in this sense means a second player who is using their own UCFK4 kit to play an interactive game such as "Pong". Communication between players is performed using the infrared serial interface.

repository. Better still, I will be able to compile your game using your Makefile that provides relative (from your build directory) dependences.

6. Ensure that you have a working Makefile that will build your program. Note that a template is provided. Also note that marks will be lost if I cannot build your application by running make.
7. Provide a readme.txt file with instructions on how to make your assignment application.
8. **Commit your source modules to your git repository by 5.00 pm Friday 20th October**.

4. Assessment

Your programs will be compiled, checked for compiler warnings and errors, and checked for plagiarism. I will award marks based on the following:

1. The modularity of your programs. I expect you to write at least one C module (both the implementation and header module), i.e., this will not simply be a string of API functions.
2. The readability of your programs (consistent formatting, commenting/documentation, avoidance of embedded constants <magic numbers>, consistent naming, etc.).
3. How well your program works during a short inspection (a rubric will be provided).
4. How simple your program is to understand. I prefer a simple well structured, well formatted, easy to read program.
5. The marking rubrics for both the demonstration and source code are available on Learn. The demonstration and source code are weighted equally (ie both are worth 5% of your final grade).

6. Plagiarism

Every year we detect plagiarism in this assignment. We use state-of-the-art systems to check your source code with material on the web (github etc.), and of course other groups, and even go back through ENCE260 eng-git repositories over several years. Any plagiarism will result in zero marks being awarded for this assignment. You may use other code (such as the code provided) if you acknowledge who wrote it. In simple terms, you may not use any part of any other group's code.

7. Suggestions

Programming is fun but not if you leave it to the last minute. Every year I hear that the hardest part about programming is getting started - so get started, now!

Look at the example apps and the header examples, i.e., pacer.h, button.h, tinygl.h, ir serial.h, etc.

1. Look at the on-line documentation on Learn.
2. Set up the preferences of your text editor to replace tabs with spaces and to indent by four characters.
3. Write a number of simple test applications, using one of the simple example applications (such as updown2) as a template.
4. Follow our coding style for ENCE260. This is documented on Learn.
5. You don't have a lot of memory (RAM or EEPROM) to play with, so keep things simple.
6. Ensure you commit your game regularly to your git repository.

7. Documentation

Documentation for the UCFK4 can be found at:
http://ecewiki.elec.canterbury.ac.nz/mediawiki/index.php/UCFK4

If you find an error, please let me know or preferably edit the documentation.