EGH456 Embedded Systems

Laboratory Exercise 1

Introduction to Embedded Systems Programming

Learning Objectives

After completing this laboratory, and following up using documentation you should be able to:

- Use Texas Instruments Tiva C Series boards to study embedded application programs
- Use Code Composer Studio for application development
- Compile, run and debug *C* programs
- Modify and run a sample program
- Understand the embedded application development process
- Understand the relationship between hardware and software at device level
- Understand *C* macros
- Understand the two different types of hardware access – direct register level access and through the peripheral driver library

Reference Documents

The textbook for this unit is Fundamentals of Embedded Software with the ARM Cortex-M3. It will be the source for theoretical concepts and the general process of embedded systems programming using *C* and *assembly*. However, it is based around the Cortex M3 and does not discuss the particular hardware platform used in this unit. The hardware and software development platforms you are using are better. In order to get maximum benefit from your learning experiences you will need to understand this limitation and consult technical documents for the laboratory component of the unit. Other useful textbooks are:

- *Real-Time Interfacing to ARM Cortex-M Microcontrollers* by Janathan W. Valvano (vol. 2 2014)

- *Fundamentals of Embedded Software – Where C and Assembly meet*, by Daniel W. Lewis, Prentice Hall

The ARM Cortex-M3 is specifically designed for embedded real-time applications and 32 bit processors now account for more than 75% of new embedded applications.

The Cortex-M4 is a later product and the TI Tiva C series EK-TM4C1294XL boards used in the laboratory component of this unit have LEDs, push button switches, USB interface, Ethernet interface etc. These boards also come with application specific *booster pack* plugin modules. These allow for easy prototyping with a broad range of applications, including capacitive touch, wireless sensing, LED lighting control, and more. There are useful online resources (video tutorials) that can be accessed from Code Composer Studio.

Information on the processor and the application development system can be found from technical documents provided by the vendor. These have been placed on the Blackboard site for the unit. The following documents are particularly useful to start with:

- Code Composer Studio Guide
- Cortex M4 Technical Reference Manual
- EK-TM4C1294XL Development Board User's Guide
- TM4C1294NCPDT Microcontroller Datasheet
- TivaWare Peripheral Driver Library User's Guide

Many books on *C* programming are available from the QUT library. For example:

- *C Programming for the absolute beginner* by Michael A. Vine, Permalink, 2007.
- *Further Topics in C Programming* by Gookin Dan, Permalink, 2015

Preparation Activities

Make sure that you can identify and use the following at your workstation in the laboratory.

- The Tiva C Launchpad development boards.
- The host PC and its USB connection to the board.
- Software installed on the host PC – including Code Composer Studio.

Open the key technical documents previously mentioned either from the links and browse through them such that you are aware of their contents and can look up necessary information when required. These documents are also available on Blackboard under Learning Resources → Labs → Lab1.

Laboratory Tasks

Open Code Composer Studio. Do **NOT** hit enter to the default work space. To make your projects portable amongst different computers, we recommend using a standard workspace name. Call the workspace **H:\egh456_ws** and hit enter.

Go to the Welcome page Note that there are many resources for learning including tutorial videos and example projects. Check out the tutorial video on Getting started. Note that the Getting Started page has links to other videos, support centre, examples, App center etc.

Learn about the Edit and Debug perspective configurations. Visit the App Center and Resource Explorer.

Click on "Browse Examples" from the Welcome page. Select the development board "EK-TM4C1294XL" from the dropdown box on the left side panel. Navigate to the example projects directory under "/Software/TM4C. . . /Dev Tools. . . /EK-TM4C. . . /Examples/". You should find the example projects required for the tasks below.

Note: If a CCS project exists on the filesystem – you may have saved to your directory or a USB drive, it can be 'imported' into the current workspace. Check the pull down menu items from 'Project'. You can switch workspaces. Check the pull down menu items from 'File'. You don't need to do this now.

Task A (Not Assessed)

Download, build and run the *blinky* example on the target board. You have to select the Project, then from the pull down menu select 'Build project'. After it has compiled, go to the Debug menu item and select Debug. To run the program click on the green arrow button.

Note: This program includes uses the *Software Driver Model* as opposed to the *Direct Register Access Model.* Section two of the TivaWare Peripheral Driver Library User's Guide provides insight into how the two models are different.

Explore the project folder for this example and find out the header files and libraries that are included. Open the *C* source file.

1. Find out the length of the data types used in the GPIO functions you see in the example program. (Check definitions in the relevant included header file)
2. Which port and which bit are used to connect the LED? (Check the source code comments)
3. What are the addresses used for data for this port? (Check the relevant included header file)
4. How is a time delay created between the LED ON and LED OFF states in this program? (Check the source code)

Task B (Not Assessed)

From within Code Composer Studio's Resource Explorer, navigate to the EK-TM4C1294XL board. From the *software* folder, find the *grlib_demo* project within the *Add-On Board Examples* folder. Build and run project.

Explore the project folder and examine the header files, the driver libraries and the *c* source code.

Task C (Assessed: 2 Marks, Deadline: Week 2)

1. Modify the *grlib_demo* program to change the banner text from "grlib demo" to your name and student number. For example: "Ashley n9123456". Build, run and test this. You will need to find the function that can do this and change a parameter as required. **Two Marks**

Show the source code and demonstrate to the tutor.

Task D (Assessed: 3 Marks, Deadline: Week 3)

Open the *blinky* example. Replace the *main* function and the includes with the code in Code Listing 1. A *copy* friendly version of this function can be downloaded from the Lab 1 Blackboard folder.

This code uses the *Direct Register Access Model* as opposed to the *Software Driver Model*. Run the program and understand what the code is doing.

1. Using the *Direct Register Access Model*, modify the program such that when *LED D1* is on, *LED D2* is off, and when *LED D2* is on, *LED D1* is off.

The TM4C1294NCPDT Microcontroller Datasheet, section 5 and 10, as well as EK-TM4C1294XL Development Board User's Guide section 2.1.5 will be particularly useful for this exercise.

Build, run and test it. Show the source code and demonstrate to the tutor.

```c
#include <stdint.h>
#include "inc/tm4c1294ncpdt.h"

int main(void)
{
    volatile uint32_t ui32Loop;

    // Enable the GPIO port that is used for the on-board LED.
    SYSCTL_RCGCGPIO_R = SYSCTL_RCGCGPIO_R12;

    // Short delay to allow peripheral to enable
    for(ui32Loop = 0; ui32Loop < 200000; ui32Loop++) { }

    // Enable GPIO pin for LED D2 (PN0)
    GPIO_PORTN_DIR_R = 0x01;

    // Set the direction as output, and enable the GPIO pin for digital
    function.
    GPIO_PORTN_DEN_R = 0x01;

    // Loop forever.
    while(1)
    {

        // Turn on the LED.
        GPIO_PORTN_DATA_R |= 0x01;

        // Delay for a bit.
        for(ui32Loop = 0; ui32Loop < 200000; ui32Loop++) { }

        // Turn off the LED.
        GPIO_PORTN_DATA_R &= ~(0x01);

        // Delay for a bit.
        for(ui32Loop = 0; ui32Loop < 200000; ui32Loop++) { }
    }
}
```

Code Listing 1: Direct Register Access for *blinky*

Assessment

Show evidence of having performed the modifications and run the programs as required to the tutor. Answers to questions must also be written down and shown in class. You may work in a group of 2 (or 3 with permission from the tutor if there is a need for it). Each member must be present in the laboratory at the workstation and be participating in the exercise to receive credit.