**Getting Started with the TivaTM TM4C123G LaunchPad Workshop**

***Student Guide and Lab Manual***

***Revision 1.22 November 2013***

**Technical Training Organization**

*Important Notice*

**Important Notice**

Texas Instruments and its subsidiaries (TI) reserve the right to make changes to their products or to discontinue any product or service without notice, and advise customers to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgment, including those pertaining to warranty, patent infringement, and limitation of liability.

TI warrants performance of its semiconductor products to the specifications applicable at the time of sale in accordance with TI’s standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Customers are responsible for their applications using TI components.

In order to minimize risks associated with the customer’s applications, adequate design and operating safeguards must be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance or customer product design. TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used. TI’s publication of information regarding any third party’s products or services does not constitute TI’s approval, warranty or endorsement thereof.

Copyright © 2013 Texas Instruments Incorporated

**Revision History**

May 2013 – Revision 1.00 Initial release May 2013 – Revision 1.01 errata May 2013 – Revision 1.02 errata May 2013 – Revision 1.03 errata June 2013 – Revision 1.04 errata July 2013 – Revision 1.10 Added Sensor Hub chapter July 2013 – Revision 1.11 errata August 2013 – Revision 1.12 Added security slide and errata August 2013 – Revision 1.20 Added PWM chapter, updated labs to TivaWare 1.1, errata October 2013 – Revision 1.21 CCS 5.5 and TivaWare 1.1 additional changes November 2013 – Revision 1.22 minor errate

**Mailing Address**

Texas Instruments Training Technical Organization 6550 Chase Oaks Blvd Building 2 Plano, TX 75023

*ii Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop*

*Table of Contents*

**Table of Contents**

**Introduction to the ARM® CortexTM-M4F and Peripherals .................. 1-1**

**Code Composer Studio ............................................................... 2-1**

**Hints and Tips ..................................................................... 2-34**

**Introduction to TivaWareTM, Initialization and GPIO ......................... 3-1**

**Interrupts and the Timers ........................................................... 4-1**

**ADC12 .................................................................................... 5-1**

**Hibernation Module .................................................................. 6-1**

**USB ....................................................................................... 7-1**

**Memory .................................................................................. 8-1**

**Floating-Point ........................................................................... 9-1**

**BoosterPacks and Graphics Library ..............................................**

**Synchronous Serial Interface .......................................................**

**UART ....................................................................................**

**μDMA ....................................................................................**

**Sensor Hub ...............................................................................**

**PWM .......................................................................................**

**10-1**

**11-1**

**12-1**

**13-1**

**14-1**

**15-1**

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - iii*

**LaunchPad Board Schematics ...................................................... Appendix**

*Table of Contents*

*iv Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop*

**Introduction**

**Introduction**

This chapter will introduce you to the basics of the Cortex-M4F and the TivaTM C Series peripherals. The lab will step you through setting up the hardware and software required for the rest of the workshop.

The Wiki page for this workshop is located here:

http://www.ti.com/TM4C123G-Launchpad-Workshop

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction 1 - 1*

**Agenda**

**Introduction to ARM® CortexTM-M4F and Peripherals** Code Composer Studio Introduction to TivaWareTM, Initialization and GPIO Interrupts and the Timers ADC12 Hibernation Module USB Memory and Security Floating-Point BoosterPacks and grLib Synchronous Serial Interface UART μDMA Sensor Hub PWM

Portfolio ...

*Chapter Topics*

**Chapter Topics**

**Introduction ................................................................................................................................................1-1**

*Chapter Topics .........................................................................................................................................1-2*

*TI Processor Portfolio and Tiva C Series Roadmap ................................................................................1-3*

*TivaTM TM4C123G Series Overview........................................................................................................1-4*

*TM4C123GH6PM Specifics .....................................................................................................................1-5*

*LaunchPad Board ....................................................................................................................................1-8*

*Lab1: Hardware and Software Set Up .....................................................................................................1-9* Objective..............................................................................................................................................1-9 Procedure ...........................................................................................................................................1-10

*1 - 2 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction*

*TI Processor Portfolio and Tiva C Series Roadmap*

**TI Processor Portfolio and Tiva C Series Roadmap**

**TI Embedded Processing Portfolio**

TM4C123G MCU ...

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction 1 - 3*

*TivaTM TM4C123G Series Overview*

**TivaTM TM4C123G Series Overview**

**TivaTM TM4C123G Microcontroller**

**Low power consumption** ◆As low as 370 μA/MHz ◆500μs wakeup from low-power modes ◆RTC currents as low as 1.7μA ◆Internal and external power control

Core and FPU ...

*1 - 4 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction*

**M4 Core and Floating-Point Unit**

◆ **32-bit ARM® CortexTM-M4 core** ◆ **Thumb2 16/32-bit code: 26% less memory & 25 % faster than pure 32-bit** ◆ **System clock frequency up to 80 MHz** ◆ **100 DMIPS @ 80MHz** ◆ **Flexible clocking system** ◆ **Internal precision oscillator** ◆ **External main oscillator with PLL support** ◆ **Internal low frequency oscillator** ◆ **Real-time-clock through Hibernation module** ◆ **Saturated math for signal processing** ◆ **Atomic bit manipulation. Read-Modify-Write using bit-banding** ◆ **Single Cycle multiply and hardware divider** ◆ **Unaligned data access for more efficient memory usage** ◆ **IEEE754 compliant single-precision floating-point unit** ◆ **JTW and Serial Wire Debug debugger access**

◆ ETM (Embedded Trace Macrocell) available through Keil and IAR emulators

Memory ...

*TM4C123GH6PM Specifics*

**TM4C123GH6PM Specifics**

**TM4C123GH6PM Memory**

**256KB Flash memory**

◆ **Single-cycle to 40MHz** ◆ **Pre-fetch buffer and speculative branch improves**

**performance above 40 MHz 32KB single-cycle SRAM with bit-banding Internal ROM loaded with TivaWare software**

◆ **Peripheral Driver Library** ◆ **Boot Loader**

0x00000000 Flash

◆ **Advanced tables**

**Encryption Standard (AES) cryptography**

0x01000000 ROM

◆ **Cyclic Redundancy Check (CRC) error**

0x20000000 SRAM **detection functionality 2KB EEPROM (fast, saves board space)**

0x22000000 Bit-banded SRAM

◆ **Wear-leveled 500K program/erase cycles**

0x40000000 Peripherals & EEPROM

◆ **Thirty-two 16-word blocks** ◆ **Can be bulk or block erased**

0x42000000 Bit-banded Peripherals

◆ **10 year data retention**

0xE0000000 Instrumentation, ETM, etc.

◆ **4 clock cycle read time**

Peripherals ...

**TM4C123GH6PM Peripherals**

**Battery-backed Hibernation Module**

◆ Internal and external power control (through external voltage regulator) ◆ Separate real-time clock (RTC) and power source ◆ VDD3ON mode retains GPIO states and settings ◆ Wake on RTC or Wake pin ◆ Sixteen 32-bit words of battery backed memory ◆ 5 μA Hibernate current with GPIO retention. 1.7 μA without **Serial Connectivity**

◆ USB 2.0 (OTG/Host/Device) ◆ 8 - UART with IrDA, 9-bit and ISO7816 support ◆ 6-I2C ◆ 4 - SPI, Microwire or TI synchronous serial interfaces ◆ 2 - CAN

More ...

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction 1 - 5*

*TM4C123GH6PM Specifics*

**TM4C123GH6PM Peripherals**

**Two 1MSPS 12-bit SAR ADCs**

◆ Twelve shared inputs ◆ Single ended and differential measurement ◆ Internal temperature sensor ◆ 4 programmable sample sequencers ◆ Flexible trigger control: SW, Timers, Analog comparators, GPIO ◆ VDDA/GNDA voltage reference ◆ Optional hardware averaging ◆ 3 analog and 16 digital comparators ◆ μDMA enabled **0 - 43 GPIO**

◆ Any interrupt GPIO can be an external edge or level triggered ◆ Can directly initiate an ADC sample sequence or μDMA transfer ◆ Toggle High-Performance rate up to the Bus CPU clock speed on the Advanced ◆ 5-V-tolerant (except for PB0/1 in input and configuration USB data pins when configured as GPIO) ◆ Programmable Drive Strength (2, 4, 8 mA or 8 mA with slew rate control) ◆ Programmable weak pull-up, pull-down, and open drain

More ...

**TM4C123GH6PM Peripherals**

**Memory Protection Unit (MPU)**

◆ Generates a Memory Management Fault on incorrect access to region **Timers**

◆ 2 Watchdog timers with separate clocks ◆ SysTick timer. 24-bit high speed RTOS and other timer ◆ Six 32-bit and Six 64-bit general purpose timers ◆ PWM and CCP modes ◆ Daisy chaining ◆ User enabled stalling on CPU Halt flag from debugger for all timers **32 channel μDMA**

◆ Basic, Ping-pong and scatter-gather modes ◆ Two priority levels ◆ 8,16 and 32-bit data sizes ◆ Interrupt enabled

More...

*1 - 6 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction*

*TM4C123GH6PM Specifics*

**TM4C123GH6PM Peripherals**

**Nested-Vectored Interrupt Controller (NVIC)**

◆ 7 exceptions and 71 interrupts with 8 programmable priority levels ◆ Tail-chaining and other low-latency features ◆ Deterministic: always 12 cycles or 6 with tail-chaining ◆ Automatic system save and restore

**Two Motion Control modules. Each with:** ◆ 8 high-resolution PWM outputs (4 pairs) ◆ H-bridge dead-band generators and hardware polarity control ◆ Fault input for low-latency shutdown ◆ Quadrature Encoder Inputs (QEI) ◆ Synchronization in and between the modules

Board...

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction 1 - 7*

*LaunchPad Board*

**LaunchPad Board**

**TivaTM EK-TM4C123GXL LaunchPad**

◆ **ARM64-pin ® CortexTM-M4F 80MHz TM4C123GH6PM** ◆ **On-board (In-Circuit USB Debug ICDI Interface)** ◆ **Micro AB USB port** ◆ **Device/ICDI power switch** ◆ **BoosterPack legacy BoosterPack XL pinout pinout also supports** ◆ **2 (SW2 user is pushbuttons connected to the WAKE pin)** ◆ **Reset button** ◆ **3 user LEDs (1 tri-color device)** ◆ **Current measurement test points** ◆ **16MHz Main Oscillator crystal** ◆ **32kHz Real Time Clock crystal** ◆ **3.3V regulator** ◆ **Support for multiple IDEs:**

*1 - 8 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction*

Lab...

*Lab1: Hardware and Software Set Up*

**Lab1: Hardware and Software Set Up**

**Objective**

The objective of this lab exercise is to download and install Code Composer Studio, as well as download the various other support documents and software to be used with this workshop. Then we’ll review the contents of the evaluation kit and verify its operation with the pre-loaded quickstart demo program. These development tools will be used throughout the remaining lab exercises in this workshop.

**Lab 1: Hardware and Software Setup**

◆ Install the software ◆ Review the kit contents ◆ Connect the hardware ◆ Test the QuickStart application

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction 1 - 9*

USB Emulation Connection

Agenda ...

*Lab1: Hardware and Software Set Up*

**Procedure**

***Hardware***

1. You will need the following hardware:

• A 32 or 64-bit Windows XP, Windows7 or 8 laptop with 2G or more of free hard drive space. 1G of RAM should be considered a minimum ... more is better. Apple laptops running any of the above OS’s are acceptable. Linux laptops are not recommended.

• Wi-Fi is highly desirable

• If you are working the labs from home, a second monitor will make the process much easier. If you are attending a live workshop, you are welcome to bring one.

• If you are attending a live workshop, **please bring a set of earphones or ear- buds.**

• If you are attending a live workshop, you will receive an evaluation board; otherwise you need to purchase one.

• If you are attending a live workshop, a digital multi-meter will be provided; otherwise you need to purchase one to complete lab 6.

• If you are attending a live workshop, you will receive a second **A-male to micro- B-male** USB cable. Otherwise, you will need to provide your own to complete Lab 7.

• If you are attending a live workshop, you will receive a Kentec 3.5” TFT LCD Touch Screen BoosterPack (**Part# EB-LM4F120-L35).** Otherwise, you will need to provide your own to complete lab 10.

• Modified Olimex 8x8 LED array Boosterpacks SensorHubs and modified R/C servos will be available to borrow during the live workshop. Otherwise you will need to purchase and modify as covered in labs 11, 14 and 15.

**As you complete each of the following steps, check the box in the title to assure that you have done everything in order.**

*1 - 10 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction*

*Lab1: Hardware and Software Set Up*

***Download and Install Code Composer Studio*** □

2. ► Download and start the latest version of Code Composer Studio (CCS) 5.x web

installer from http://processors.wiki.ti.com/index.php/Download\_CCS (do not download any beta versions). Bear in mind that the web installer will require Internet access until it completes. If the web installer version is unavailable or you can’t get it to work, download, unzip and run the offline version. The offline download will be much larger than the installed size of CCS since it includes all the possible supported hardware.

This version of the workshop was constructed using CCS version 5.5. Your version may be later. For this and the next few steps, you will need a my.TI account (you will be prompted to create one or log into your existing account).

Note that the “free” license of CCS will operate with full functionality for free while connected to a TivaTM C Series evaluation board.

3. If you downloaded the offline file, ► launch the ccs\_setup\_5.xxxxx.exe file in

the folder created when you unzipped the download.

4. ► Accept the Software License Agreement and click Next.

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction 1 - 11*

*Lab1: Hardware and Software Set Up*

5. Unless you have a specific reason to install CCS in another location, ► accept the default installation folder and ► click Next. If you have another version of CCS and you want to keep it, we recommend that you install this version into a different folder.

6. ► Select “Custom” for the Setup type and click Next.

*1 - 12 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction*

*Lab1: Hardware and Software Set Up*

7. In the next dialog, ► select the processors that your CCS installation will support. You must select “Tiva C Series ARM MCUs” in order to run the labs in this workshop. You can select other architectures, but the installation time and size will increase.

► Click Next.

8. In the Component dialog, keep the default selections and ► click Next.

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction 1 - 13*

*Lab1: Hardware and Software Set Up*

9. In the Emulators dialog, ► uncheck the Blackhawk and Spectrum Digital emulators,

unless you plan on using either of these. ► Click Next.

10. When you reach the final installation dialog, ► click Next. The web installer process

should take 15 - 30 minutes, depending on the speed of your connection. The offline installation should take 10 to 15 minutes. When the installation is complete, uncheck the “Launch Code Composer Studio v5” checkbox and then ► click Finish.

*1 - 14 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction*

*Lab1: Hardware and Software Set Up*

11. There are several additional tools that require installation during the CCS install process.

Click “Yes” or “OK” to proceed when these appear.

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction 1 - 15*

*Lab1: Hardware and Software Set Up*

***Install TivaWareTM for C Series (Complete)*** □

12. ► Download and install the latest full version of TivaWare

from: http://www.ti.com/tool/sw-tm4c . The filename is SW-TM4C-x.x.exe . This workshop was built using version 1.1. Your version may be a later one. If at all possible, please install TivaWare into the default C:\TI\TivaWare\_C\_Series-x.x folder.

***Install LM Flash Programmer*** □

13. ► Download, unzip and install the latest LM Flash Programmer

(LMFLASHPROGRAMMER) from http://www.ti.com/tool/lmflashprogrammer .

***Download and Install Workshop Lab Files*** □

14. ► Download and install the lab installation file from the workshop materials section of

the Wiki site below. The file will install your lab files in: C:\Tiva\_TM4C123G\_LaunchPad. http://www.ti.com/TM4C123G-Launchpad-Workshop

***Download Workshop Workbook*** □

15. ► Download a copy of the workbook pdf file from the workshop materials section of the

Wiki site below to your desktop. It will be handy for copying and pasting code.

http://www.ti.com/TM4C123G-Launchpad-Workshop

***Terminal Program*** □

16. If you are running WindowsXP, you can use HyperTerminal as your terminal program.

Windows7 does not have a terminal program built-in, but there are many third-party alternatives. The instructions in the labs utilize HyperTerminal and PuTTY. You can download PuTTY from the address below.

http://the.earth.li/~sgtatham/putty/latest/x86/putty.exe

***Windows-side USB Examples*** □

17. ► Download and install the Windows-side USB examples from this site:

www.ti.com/sw-usb-win

***Download and Install GIMP*** □

18. We will need a graphics manipulation tool capable of handling PNM formatted images.

GIMP can do that. ► Download and install GIMP from here: www.gimp.org

*1 - 16 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction*

*Lab1: Hardware and Software Set Up*

***LaunchPad Board Schematic***

19. For your reference, the schematic is included at the end of this workbook.

***Helpful Documents and Sites***

20. There are many helpful documents that you should have, but at a minimum you should

have the following documents at your fingertips.

With TivaWareTM installed, look in C:\TI\TivaWare\_C\_Series-1.1\docs and you’ll find:

**Peripheral Driver User’s Guide (SW-DRL-UG-x.x.pdf)**

**USB Library User’s Guide (SW-USBL-UG-x.x.pdf)**

**Graphics Library User’s Guide (SW-GRL-UG-x.x.pdf)**

**LaunchPad Firmware User’s Guide (SW-EK-TM4C123GXL-UG-x.x.pdf )**

21. Go to: http://www.ti.com/lit/gpn/tm4c123gh6pm and download the TM4C123GH6PM

Microcontroller Data Sheet. TivaTM C Series data sheets are actually the complete user’s guide to the device, so expect a large document.

22. If you are migrating from an earlier Stellaris design, you will find this document

ful: http://www.ti.com/litv/pdf/spma050a

23. Download the ARM Optimizing C/C++ Compilers User Guide

from http://www.ti.com/lit/pdf/spnu151 (SPNU151). Of particular interest are the sizes for all the different data types in table 6-2. You may see the use of “TMS470” here ... that is the TI product number for its ARM devices.

24. You will find a “Hints” section at the end of chapter 2. This information will be handy if

you run into problems during the labs.

***You can find additional information at these websites:***

**Main page:** www.ti.com/launchpad

**Tiva C Series TM4C123G LaunchPad:** http://www.ti.com/tool/ek-tm4c123gxl

**TM4C123GH6PM folder:** http://www.ti.com/product/tm4c123gh6pm

**BoosterPack webpage:** www.ti.com/boosterpack

**LaunchPad Wiki:** www.ti.com/launchpadwiki

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction 1 - 17*

*Lab1: Hardware and Software Set Up*

***Kit Contents***

**25.** ► **Open up your kit**

You should find the following in your box:

• **The TM4C123GXL LaunchPad Board**

• **USB cable (A-male to micro-B-male)**

• **README First card**

• **If you are in a live workshop, you should find a 2nd USB cable**

***Initial Board Set-Up***

**26. Connecting the board and installing the drivers**

The TM4C123GXL LaunchPad Board ICDI USB port (marked DEBUG and shown in the picture below) is a composite USB port and consists of three con- nections:

**Stellaris ICDI JTAG/SWD Interface** - debugger connection **Stellaris ICDI DFU Device** - firmware update connection **Stellaris Virtual Serial Port** - a serial data connection

Using the included USB cable, ► connect the USB emulation connector on your evalu- ation board (marked DEBUG) to a free USB port on your PC. A PC’s USB port is capable of sourcing up to 500 mA for each attached device, which is sufficient for the evaluation board. If connecting the board through a USB hub, it must be a powered hub.

The drivers should install automatically. If they do not, the steps to install them will be covered shortly.

*1 - 18 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction*

*Lab1: Hardware and Software Set Up*

***QuickStart Application***

Your LaunchPad Board came preprogrammed with a quickstart application. Once you have powered the board, this application runs automatically. You probably already no- ticed it running as you installed the drivers.

27. Make sure that the power switch in the upper left hand cor-

ner of your board is in the right-hand DEBUG position as shown:

28. The software on the TM4C123GH6PM uses the timers as

pulse-width modulators (PWMs) to vary the intensity of all three colors on the RGB LED (red, green, and blue) individually. By doing so, your eye will perceive many different colors created by combining those primary colors.

The two pushbuttons at the bottom of your board are marked **SW1** (the left one) and **SW2** (the right one). ► Press or press and hold **SW1**to move towards the red- end of the color spectrum. ► Press or press and hold **SW2** to move towards the violet-end of the color spectrum.

If no button is pressed for 5 seconds, the software returns to automatically chang- ing the color display.

29. ► Press and hold both **SW1** and **SW2** for 3 seconds to enter hibernate mode. In

this mode the last color will blink on the LEDs for 1⁄2 second every 3 seconds. Be- tween the blinks, the device is in the VDD3ON hibernate mode with the real- time-clock (RTC) running. ► Pressing **SW2** at any time will wake the device and return to automatically changing the color display.

30. We can communicate with the board through the UART. The UART is connected

as a virtual serial port through the emulator USB connection.

The following steps will show how to open a connection to the board using HyperTerminal (in WinXP) and PuTTY (in Windows 7 or 8).

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction 1 - 19*

*Lab1: Hardware and Software Set Up*

31. We need to find the COM port number of the Stellaris Virtual Serial Port in the

Device Manager. **Skip to step 32 if you are using Windows 7 or 8.**

**Windows XP:**

A. ► Click on the Windows Start button. ► Right-click on My Computer and se- lect Properties from the drop-down menu.

B. In the System Properties window, ► click the Hardware tab.

C. ► Click the Device Manager button.

The Device Manager window displays a list of hardware devices installed on your computer and allows you to set the properties for each device. If you see any of the three devices listed in step 26 in the “Other” category, it means that the driver for those devices is not installed. Run step 37, and then return to here.

► Expand the Ports heading and write number for the Stellaris Virtual Serial Port here: **COM\_\_\_\_\_**

*1 - 20 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction*

*Lab1: Hardware and Software Set Up*

**32. Windows 7 or 8:**

A. ► Click on the Windows Start button. ► Right-click on Computer and select

Properties from the drop-down menu.

B. ► Click on Device Manager on the left of the dialog.

The Device Manager window displays a list of hardware devices installed on your computer and allows you to set the properties for each device. If you see any of the three devices listed in step 26 in the “Other” category, it means that the driver for those devices is not installed. Run step 37, and then return to here.

► Expand the Ports heading and write number for the Stellaris Virtual Serial Port here: **COM\_\_\_\_\_**

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction 1 - 21*

*Lab1: Hardware and Software Set Up*

33. **In WinXP**, open HyperTerminal by ► clicking Start → Run..., then type

hypertrm in the Open: box and click OK. Pick any name you like for your connection and click OK. In the next dialog box, change the Connect using: selection to COM##, where ## is the COM port number you noted earlier. Click OK. Make the selections shown below and click OK.

When the terminal window opens, press Enter once and the LaunchPad board will respond with a > indicating that communication is open. Skip to step 31.

34. In **Win7 or 8**, ► double-click on putty.exe. Make the settings shown below

and then click Open. Your COM port number will be the one you noted earlier

When the terminal window opens, press Enter once and the LaunchPad board will respond with a > indicating that communication is open.

*1 - 22 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction*

*Lab1: Hardware and Software Set Up*

35. You can communicate by ► typing the following commands and pressing enter:

**help:** will generate a list of commands and information

**hib:** will place the device into hibernation mode. Pressing SW2 will wake the

device.

**rand:** will start a pseudo-random sequence of colors

**intensity:** adjust the LED brightness between 0 to 100 percent. For instance intensity 100 will change the LED to maximum brightness.

**rgb:** follow with a 6 hex character value to set the intensity of all three LEDs. For instance: rgb FF0000 lights the red LED, rgb 00FF00 lights the blue LED and rgb 0000FF lights the green LED.

36. ► Close your terminal program.

You’re done.

37. **Run this step only if your device drivers did not install properly.**

► Obtain the ICDI drivers from your instructor or download the zip file from http://www.ti.com/tool/stellaris\_icdi\_drivers. ► Unzip the file to a folder on your desktop. ► Back in the Device Manager, right-click on each of the “Other” devices (one at the time) and select Update Driver. In the following dialogs point the wizard to the folder on your desktop with the unzipped files.

If the process seems to take longer than it should, the wizard is likely searching on-line. Turn off your wireless or disconnect your network cable to prevent this.

► Make sure all three devices listed in step 26 are properly installed.

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction 1 - 23*

*Lab1: Hardware and Software Set Up*

*1 - 24 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Introduction*

**Code Composer Studio**

**Introduction**

This chapter will introduce you to the basics of Code Composer Studio. In the lab, we will explore some Code Composer features.

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio 2 - 1*

**Agenda**

Introduction to ARM® CortexTM-M4F and Peripherals **Code Composer Studio** Introduction to TivaWareTM, Initialization and GPIO Interrupts and the Timers ADC12 Hibernation Module USB Memory and Security Floating-Point BoosterPacks and grLib Synchronous Serial Interface UART μDMA Sensor Hub PWM

IDEs...

*Chapter Topics*

**Chapter Topics**

**Code Composer Studio ..............................................................................................................................2-1**

*Chapter Topics .........................................................................................................................................2-2*

*Tiva C Series Development Tools ............................................................................................................2-3*

*TI Software and Ecosystem ......................................................................................................................2-4*

*Code Composer Studio Functional Overview ..........................................................................................2-5*

*Target Configuration and Emulators .......................................................................................................2-6*

*Projects and Workspaces .........................................................................................................................2-7*

*Creating a New Project and Adding Files ...............................................................................................2-8*

*Portable Projects .....................................................................................................................................2-9*

*Path and Build Variables .......................................................................................................................2-10*

*Build Configurations ..............................................................................................................................2-11*

*Licensing and Pricing ............................................................................................................................2-12*

*Lab2: Code Composer Studio ................................................................................................................2-13* Objective............................................................................................................................................2-13

*Lab 2 Procedure ....................................................................................................................................2-14* Add Path and Build Variables ...........................................................................................................2-18 Add files to your project ....................................................................................................................2-20 Build, Load, Run ...............................................................................................................................2-24 Perspectives .......................................................................................................................................2-26 VARS.INI – An Easier Way to Add Variables ..................................................................................2-28

*LM Flash Programmer ..........................................................................................................................2-30*

*Optional: Creating a bin File for the Flash Programmer .....................................................................2-32*

*Hints and Tips ........................................................................................................................................2-33*

*2 - 2 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio*

*Tiva C Series Development Tools*

**Tiva C Series Development Tools**

**Development Tools for Tiva C Series MCUs**

**Eval License**

**Kit Upgradeable**

**30-day function. full 32KB Upgradeable**

**limited. code size 32KB Upgradeable**

**limited. code size emulation Full Onboard function. limited**

**Compiler GNU C/C++ IAR C/C++ RealView C/C++ TI C/C++**

**Debugger IDE / gdb / Eclipse Workbench Embedded C-SPY / μVision CCS/Eclipse- based suite**

**Full Upgrade**

**full 2800 personal edition 99 support USD USD / 2700 USD** *Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio 2 - 3*

**MDK-Basic KB) (2895 = €2000 USD) (256**

**445 USD**

**Debugger JTAG J-Link, 299 USD U-Link, 199 USD XDS100, 79 USD**

TI SW Ecosystem ...

*TI Software and Ecosystem*

**TI Software and Ecosystem**

*2 - 4 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio*

**TI Software and Tools Ecosystem**

**Run-Time Software Development Tools**

• **High-level TI-RTOS OS support and** • **OS TI-Wares Independent software support packages**

**and** • **TI software, Design tools Network: and off-the-shelf services**

• **Forums & Wikis**

• **In-person and online training**

Run-Time Software ...

**TI Wares: minimizes programming complexity w/ optimized drivers & OS independent support for TI solutions**

• **Low-level driver libraries**

• **Peripheral programming interface**

• **Tool-chain agnostic C code**

• **Available today**

**Support & Community**

**TI-RTOS: provides an optimized real- time kernel at no charge Wares that works with TI** • **Real-time kernel for TI (SYSBIOS) devices: + optimized** • **Scheduling**

• **Memory management**

• **Utilities**

• **Foundational software packages (TI Wares)**

• **Libraries and examples**

• **TI RTOS available today**

**Run-Time Software**

**TI-RTOS**

**SYSBIOS +**

**TI Wares**

**+**

**SDK Software Development Kit**

CCS Functional Overview ...

• **CCStudioTM Integrated Development Environment (IDE) and other IDEs**

• **Optimizing compilers**

• **Design Kits & Evaluation Modules**

• **File systems**

• **Network stack**

• **USB**

*Code Composer Studio Functional Overview*

**Code Composer Studio Functional Overview**

**Code Composer Studio Functional Overview**

Compiler

**.asm**

Standard Libraries Runtime SYS/BIOS

Libraries

Target Config

**Simulator** File

**.c**

**.lib**

**.ccxml**

**Edit .asm** Assembler **.obj**

Linker

**.out**

**Debug**

**LaunchPad Emulator/** SYS/BIOS Config (.cfg) User.cmd

**.map** Bios.cmd

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio 2 - 5*

**Stand-Alone Emulator**

◆ Integrated Development Environment (IDE) based on **Eclipse** ◆ Contains debugger, all BIOS development and includes tools one – compilers, target – the assembler, Simulator linker, ◆ GEL memory, files initialize peripherals, the debugger etc. are

so that it understands where **... .gel**

Target Board

Target configuration and Emulators...

*Target Configuration and Emulators*

**Target Configuration and Emulators**

**Target Configuration and Emulators**

◆ The Target Configuration File specifies:

• Connection to the target (Simulator or Emulator type)

• Target device

• GEL file (if applicable) for hardware setup

◆ Emulator (Connection) Options

• Built-in Spectrum and Digital external and emulators others from TI, Blackhawk, • XDS100v1/v2, 200, 510, 560, 560v2

*2 - 6 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio*

Projects and Workspaces ...

*Projects and Workspaces*

**Projects and Workspaces**

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio 2 - 7*

Link

Link

Link

◆ WORKSPACE folder contains:

• IDE settings and preferences

• Projects can ***reside in*** the workspace folder or be ***linked*** from elsewhere

• When importing projects into the workspace, linking is recommended

• Deleting a project within the Project Explorer only deletes the link

Creating a New Project ...

**Source**

Projects and Workspaces ...

**Projects and Workspaces**

**Workspace**

• *Project 1*

• *Project 2*

• *Project 3*

• *Settings/preferences*

◆ PROJECT folder contains:

• Build in managed and tool MAKE settings projects) (for use • Files ***reside*** can ***in*** be the ***linked*** project ***to*** folder or • Deleting Project the link

Explorer a linked only file within deletes the

**Projects and Workspaces (viewed in CCS)**

Link

**Project**

• *Source Files*

• *Header Files*

• *Library Files*

• *Build/tool settings*

**PROJECT**

**WORKSPACE**

**Source Files**

• *Code and Data*

**Header Files**

• *Declarations*

**Library Files**

• *Code and Data*

*Creating a New Project and Adding Files*

**Creating a New Project and Adding Files**

*2 - 8 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio*

**Creating a New Project**

**File New CCS Project**

*(in Edit perspective...)*

◆ **Project Location**

• Default = workspace

• Manual = anywhere you like ◆ **Connection**

• If choose target target configuration “connection” is specified, file) user (i.e. can the ◆ **Project templates**

• Empty

• Empty but with a main.c

• Assembly only

• BIOS

• others

Adding Files to a Project ...

**Adding Files to a Project**

◆ **Users can ADD (copy or link) files into their project**

• SOURCE files are typically COPIED

• LIBRARY files are typically LINKED (referenced)

**1** Right-click on project and select: **2** Select file(s) to add to the project:

**3** Select “Copy” or “Link” ◆ **COPY**

• Copies to *project* file *folder* from original (two copies) location ◆ **LINK**

• References file in the *original* (points *folder* to) source • Can typically select PROJECT\_LOC

a “reference” point –

Making a Project Portable ...

*Portable Projects*

**Portable Projects**

**Portable Projects**

◆ **Why make your projects “portable”?**

• Simplifies project sharing

• You can easily re-locate your projects

• Allow simple changes to link to new releases of software libraries

Copied move with files the are project not a problem folder) (they Linked are folder located via files a: may outside be an the issue. project They • absolute path, or

• relative path

This for can linked a be is relative file. the specified Path path. Variable for This every *Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio 2 - 9*

Path and Build Variables ...

*Path and Build Variables*

**Path and Build Variables**

**Path Variables and Build Variables**

◆ **Path Variables**

• Used by CCS (Eclipse) to store the base path for relative linked files

• Example: PROJECT\_LOC is set to the path of your project, say

c:/Tiva\_LaunchPad\_Workshop/lab2/project

• Used as a reference point for relative paths, e.g.

${PROJECT\_LOC}/../files/main.c

◆ **Build Variables**

• Used by CCS (Eclipse) to store base path for build libraries or files

• Example: generation CG\_TOOL\_ROOT tools (compiler/linker) is set to the path for the code • Used ${CG\_TOOL\_ROOT}/include to find #include .h files, or or object ${CG\_TOOL\_ROOT}/lib

libraries, e.g.

◆ **How are these variables defined?**

• The when install variables you CCS create with in the these a new build examples project tools (CG\_TOOL\_ROOT) (PROJECT\_LOC) are automatically and defined when you • What some about new variables TivaWare yourself

or additional software libraries? You can define Adding Variables ...

**Adding Variables** ◆ **Why are we doing this?**

• We the could base for use build PROJECT\_LOC variables for all linked resources or PROJECT\_ROOT as • It to is put “almost” it at the portable, same “level” BUT if in you the move file system or copy your project, you have • Defining path that a does link and NOT build depend variable on location for TivaWare of the project location (much gives us more a relative portable)

• Also, variables if we – install which a is new much version easier of than TivaWare, creating we new only relative need to links change these ◆ **How to add Path and Build Variables**

• *Project Linked* to add *Resources*. a → new *Properties*, path You variable expand will see the a tab *Resource* for *Path* category, *Variables*, click click on *New* • *Project* Click *New* → to *Properties*, add a new click build on variable *Build* category, click on the *Variables* tab, • In to the be the lab, path we’ll of add the a latest path variable TivaWare and release build variable TIVAWARE\_INSTALL ◆ **Note:**

• This method defines the variables as part of the project (finer control)

• You can also define variables as part of your workspace (do it once)

Build Configurations ...

*2 - 10 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio*

*Build Configurations*

**Build Configurations Build Configurations**

◆ Code Composer has two pre-defined BUILD CONFIGURATIONS:

• *Debug* (symbols, no optimization) – great for LOGICAL debug

• *Release* (no symbols, optimization) – great for PERFORMANCE ◆ Users can create their own custom build configurations

• Right-click on the project and select *Properties*

• Then click “*Processor Options*” or any other category:

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio 2 - 11*

CCS Licensing and Pricing ...

*Licensing and Pricing*

**Licensing and Pricing**

**CCSv5 Licensing and Pricing**

◆ **Licensing**

• Wide variety of options (node locked, floating, time based)

• All versions (full, DSK, free tools) use the same image

• Updates readily available online

◆ **Pricing**

• Includes FREE options noted below

• Annual subscription - $99 *($159 for floating license)*

**Item Description Price Annual** Platinum Eval Tools Full tools with 90 day limit (all EMU) FREE Platinum Bundle XDS100 use (EVM or simulator) FREE \* Platinum Node Lock Full tools tied to a machine $495/$445 \*\* $99 Platinum Floating Full tools shared across machines $795 $159 MSP430 Code-Limited MSP430 (16KB code limit) FREE

\* recommended option: purchase Development Kit, use XDS100v1-2, & Free CCSv5 \*\* $495 includes DVD, $445 is download only

CCS FYI ...

**CCSv5 – For More Information**

*2 - 12 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio*

http://processors.wiki.ti.com/index.php/Category:CCS\_Training

Lab ...

*Lab2: Code Composer Studio*

**Lab2: Code Composer Studio**

**Objective**

In this lab, we’ll create a project that contains two source files, main.c and tm4c123gh6pm\_startup\_ccs.c, which contain the code to blink an LED on your LaunchPad board. The purpose of this lab is to practice creating projects and getting to know the look and feel of Code Composer Studio. In later labs we’ll examine the code in more detail. So far now, don’t worry about the C code we’ll be using in this lab.

**Lab 2: Code Composer Studio**

◆ Create a new project ◆ Experiment with some CCS features ◆ Use the LM Flash Programmer

Agenda ...

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio 2 - 13*

USB Emulation Connection

*Lab 2 Procedure*

**Lab 2 Procedure**

***Folder Structure for the Labs***

**1. Browse the directory structure for the workshop labs**

► Using Windows Explorer, locate the following folder:

C:\TM4C123G\_LaunchPad\_Workshop

In this folder, you will find all the lab folders for the workshop. If you don’t see this folder on your c:\ drive, check to make sure you have installed the workshop lab files. Expand the \lab2 folder and you’ll notice that there are two sub-folders \files and \project. The \files folder will sometimes contain additional files for your reference. The \project folder will contain your project settings and files for both the projects that you create and the projects we created that you will import. It will also contain solution files saved as text files. You will be able to see these files in the Project Explorer and easily cut/paste the contents into your files if and when necessary.

**Note:** When you create a project, you have a choice to use the “default location” which is the

CCS workspace or to select another location. In this workshop, we will not be using the workspace for the project files; rather, we’ll use the folder where you installed the lab files, C:\TM4C123G\_LaunchPad\_Workshop.

The workspace will only contain CCS settings, and links to the projects we create or import.

*2 - 14 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio*

*Lab 2 Procedure*

***Create a New CCS Project***

**2. Create a new project**

► Launch CCS. When the “Select a workspace” dialog appears, ► browse to your My Documents folder:

(In WinXP) C:\Documents and Settings\<user>\My Documents

(In Win7 or 8) C:\Users\<user>\My Documents

Obviously, replace <user> with your own username. The name and location for your workspace isn’t critical, but we suggest that you use **MyWorkspaceTM4C123G**. Do not check the “*Use this as the default and do not ask again*” checkbox. If at some point you accidentally check this box, it can be changed in CCS.

► Click OK.

**3. Select a CCS License**

If you haven’t already licensed Code Composer, you may be asked to do so in the next few installation steps. You can do this step manually from the CCS Help menu.

► Click on *Help* → *Code Composer Studio Licensing Information*.

► Select the “*Upgrade*” tab, and then select the “*Free*” license. As long as your PC is connected to the LaunchPad board, CCS will have full functionality, free of charge.

**4. Close TI Resource Explorer and/or Grace**

When the “TI Resource Explorer” and/or “Grace” windows appear, close these windows using the “X” on the tab. At this time, these tools support other processor families, e.g. MSP430.

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio 2 - 15*

*Lab 2 Procedure*

**5. Create a New Project**

To create a new project, ► select *Project* → *New CCS Project:*

► For the project name, type *lab2*

► **Uncheck** the box “*Use default location*” and click the *Browse...* button. Navigate to: C:\TM4C123G\_LaunchPad\_Workshop\lab2\project

and click *OK*.

► Select Device family: *ARM*, for Variant, type *123G* in the filter text field, then select *Tiva TM4C123GH6PM* in the drop-down box (typing 123G narrows the list making it easier to find the exact part on the Tiva LaunchPad board.

► For Connection: choose *Stellaris In- Circuit Debug Interface* . This is the built- in emulator on the LaunchPad board.

► In the Project templates and examples box, choose *Empty Project* and then click *Finish*.

*2 - 16 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio*

*Lab 2 Procedure*

**6. Review the CCS Editing GUI**

Note the names of the Code Composer GUI panes above.

► In the Project Explorer pane on your desktop, click the symbol next to *lab2, Includes* and *targetConfigs* to expand the project. Your project should look like the above.

**7.** You probably noticed that the New Project wizard added a startup file called

tm4c123gh6pm\_startup\_ccs.c into the project automatically. We’ll look more closely at the contents of this file later.

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio 2 - 17*

**Project Explorer pane**

**Console pane**

**Editor pane**

**Problems pane**

**Problems pane**

*Lab 2 Procedure*

**Add Path and Build Variables**

If you recall in the presentation, the path and build variables are used for:

• Path variable – when you ADD (link) a file to your project, you can specify a “relative to” path. The default is *PROJECT\_LOC* which means that your linked resource (like a .lib file) will be linked relative to your project directory.

• Build variable – used for items such as the search path for include files associated with a library – i.e. it is used when you build your project.

Variables can either have a *PROJECT* scope (that they only work for this project) or a *WORKSPACE* scope (that they work across all projects in the workspace).

In the next step, we need to add (link) a library file and then add a search path for include files. First, we’ll add these variables MANUALLY as *PROJECT* variables. Later, we will show you a quick and easy way to add these variables into your *WORKSPACE* so that any project in your workspace can use the variables.

**8. Adding a Path Variable**

To add a path variable, ► Right-click on your project and select *Properties*. ► Expand the *Resource* list in the upper left-hand corner as shown and click on *Linked Resources*:

You will see two tabs on the right side – *Path Variables* and *Linked Resources:*

In the Path Variables tab, notice that *PROJECT\_LOC* is listed and will display as the default path variable for linked resources in your project.

We want to add a *New* variable to specify exactly where you installed TivaWare.

► Click *New*

► When the New Variable dialog appears, type TIVAWARE\_INSTALL for the *name*.

► For the *Location*, click the *Folder...* button and navigate to your TivaWare installation. Click on the folder name and then click OK.

► Click OK. You should see your new path variable listed in the Path Variables list.

*2 - 18 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio*

*Lab 2 Procedure*

**9. Adding a Build Variable**

Now let’s add a build variable that we will use in the include search path for the INCLUDE files associated with the TivaWare driver libraries.

► Click on *Build* and then the *Variables* tab:

► Click the *Add* button. When the *Define a New Build Variable* dialog appears, insert TIVAWARE\_INSTALL into the Variables name box.

► Check the “Apply to all configurations” checkbox

► Change the Type to Directory and browse to your Tivaware installation folder.

► Click OK.

► Click OK again to save and close the Build Properties window.

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio 2 - 19*

*Lab 2 Procedure*

**Add files to your project**

We need to add main.c to the project. We also need to add the TivaWare driverlib.lib object library. The C file should be copied to the project, the driverlib file should be linked.

**10. Add (copy) the C file**

► Select *Project* → *Add Files...* ► Navigate to the folder:

C:\TM4C123G\_LaunchPad\_Workshop\lab2\files

Select main.c and click *Open*.

Then select *Copy Files* and click *OK*.

*2 - 20 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio*

*Lab 2 Procedure*

**11. Link the TivaWare driverlib.lib file to your project**

► Select *Project-Add Files...* Navigate to:

C:\TI\TivaWare\_C\_Series-1.1\driverlib\ccs\Debug\driverlib.lib

... and ► click Open. The File Operation dialog will open ...

Use the *TIVAWARE\_INSTALL* path variable you created earlier. This means that the LINK (or reference to the library) file will be RELATIVE to the location of the TivaWare installation. If you hand this project to someone else, they can install the project anywhere in the file system and this link will still work. If you choose *PROJECT\_LOC,* you would get a path that is relative to the location of your project and it would require the project to be installed at the same “level” in the directory structure. Another advantage of this approach is that if you wanted to link to a new version, say TivaWare\_C\_Series-1.2, all you have to do is modify the variable to the new folder name.

► Make the selections shown and click OK.

Your project should now look something like the screen capture below. Note the symbol for driverlib.lib denotes a linked file.

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio 2 - 21*

*Lab 2 Procedure*

**12. Add the INCLUDE search paths for the header files**

► Open main.c by double-clicking on the filename in the Project Explorer pane of CCS. You should see “?” warnings in the left margin which indicate “unresolved inclusion”. Hover your cursor over the question mark to see the helpful message.

Until now, you haven’t told the project where to find these header files.

► Right-click on your lab2 project in the Project Explorer pane and select *Properties*.

► Click on *Build → ARM Compiler → Include Options* (as shown):

► In the **lower-right** panel, click the “**+**” sign next to *Add dir to #include search path*

and add the following path using the build variable you created earlier. Place the variable name inside **braces**, after the $ as shown:

${TIVAWARE\_INSTALL}

► Click OK.

► Click OK again, and now you should see those “?” in main.c disappear after a moment.

Problem solved.

*2 - 22 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio*

*Lab 2 Procedure*

**13. Examine your Project files using Windows Explorer**

► Using Windows Explorer, locate your lab2 project folder:

C:\TM4C123G\_LaunchPad\_Workshop\lab2\project

Do you see main.c? It should be there because you copied it there. Do you see the driverlib.lib file? This file should NOT be there because it’s only linked in your project. Notice the other folders in the \project folder – these contain your CCS project- specific settings. Close Windows Explorer.

**14. Examine the properties of your new project**

► In CCS, right-click on your project and select *Properties*. Click on each of the sections below:

**Resource**: This will show you the path of your current project and the resolved path if it is linked into the workspace. Click on “*Linked Resources*” and both tabs associated with this.

What is the PROJECT\_LOC path? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Are there any linked resources? If so, what file(s)? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**General**: shows the main project settings. Notice you can change almost every field here AFTER the project was created.

**Build → ARM Compiler**: These are the basic compiler settings along with every compiler setting for your project.

**Other**: feel free to click on a few more settings, but don’t change any of them.

► Click *Cancel*.

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio 2 - 23*

*Lab 2 Procedure*

**Build, Load, Run**

**15. Build your project and fix any errors**

► Assure that your LaunchPad is connected to your laptop. Build and load your project to the TM4C123GH6PM flash memory by clicking the Debug button . If you ever want to build the project without loading it, click the HAMMER (Build) button.

► Fix any errors that occur. For the present you can ignore any warnings. If you encounter the error shown, your board is disconnected, your power switch is in the wrong position or your drivers are incorrectly installed.

The program counter will run to main() and stop as shown:

**16. Getting to know the CCS Debug GUI**

Note the names of the Code Composer panes above. There are two pre-defined perspectives in Code Composer; CCS Edit and CCS Debug. ► Click and drag the tabs (at the arrow above) to the left so you can see both. Perspectives are only a “view” of the available data ... you can edit your code here without changing perspectives. And you can modify these or create as many additional perspectives as you like. More on that in a moment.

*2 - 24 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio*

**Debug Pane Watch & Expressions Panes**

**Code/Editor Pane**

**Console and Problems Panes**

*Lab 2 Procedure*

**17. Run your program.**

► Click the Resume button or press the F8 key on your keyboard:

The tri-color LED on your target board should blink showing the three colors in sequence. If not, attempt to solve the problem yourself for a few minutes, and then ask your instructor for help.

To stop your program running, ► click the Suspend button:

If the code stops with a “No source available ...” indication, click on the main.c tab. Most of the time in the while() loop is spent inside the delay function. That source file is not linked into this project.

**18. Set a Breakpoint**

In the code window in the middle of your screen, double-click in the blue area to the left of the line number of the GPIOPinWrite() instruction. This will set a breakpoint (it will look like

this: ). Click the Resume button to restart the code. The program will stop at the breakpoint and you will see an arrow on the left of the line number, indicating that the program counter has stopped on this line of code. **Note that the current ICDI driver does not support adding or removing breakpoints while the processor is running.** Click the Resume button a few times or press the F8 key to run the code. Observe the LED on the LaunchPad board as you do this.

**19. View/Watch memory and variables.**

► Click on the Expressions tab in the Watch and Expressions pane.

► Double-click on the ui8*LED* variable anywhere in main().

► Right-click on ui8*LED* and select:

► Click OK. Right-click on ui8*LED* in the Expressions pane, and select Number Format → Hex. Note the value of ui8*LED*.

Of course, the ui8*LED* variable is located in SRAM. You can see the address in the expressions view. But let’s go see it in memory.

► Select *View → Memory Browser:*

► Type *&ui8LED* into the memory window to display ui8*LED* in memory:

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio 2 - 25*

*Lab 2 Procedure*

**20. View Registers**

► Select *View → Registers* and notice that you can see the contents of all of the registers in your target’s architecture. This is very handy for debugging purposes.

► Click on the arrow on the left to expand the register view. Note that non-system peripherals that have not been enabled cannot be read. In this project you can view Core Registers, GPIO\_PORTA (where the UART pins are), GPIO\_PORTF (where the LEDs and pushbuttons are located), HIB, FLASH\_CTRL, SYSCTL and NVIC.

**Perspectives**

CCS perspectives are quite flexible. You can customize the perspective(s) and save them as your own custom views if you like. It’s easy to resize, maximize, open different views, close views, and occasionally, you might wonder “How do I get things back to normal?”

**21. Let’s move some windows around and then reset the perspective.**

► Right-click on the *Console* window tab and select “*Detached*”. You can now move this window around wherever you want. ► Right click again and select “Detached” to re-attach it.

In the editing pane, ► double-click on the tab showing main.c:

Notice that the editor window maximizes to full screen. Double-click on the tab again to restore it.

► Move some windows around on your desktop by clicking-and-holding on the tabs.

Whenever you get lost or some windows seem to have disappeared in either the CCS Edit, CCS Debug or your own perspectives, you can restore the window arrangement back to the default.

► Find and click the Restore button on the left or right of your display. If you want to reset the view to the factory default you can also choose *Window* → *Reset Perspective*:

**NOTE: Do not use the perspective tabs to move back and forth between perspectives.**

**Clicking the CCS Debug tab only changes the view; it does not connect to the device, download the code or start a debug session. Likewise, clicking the CCS Edit tab does not terminate a debug session.**

**Only use the Debug and Terminate buttons to move between perspectives in this workshop.**

*2 - 26 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio*

*Lab 2 Procedure*

**22. Remove all breakpoints**

► Click *Run* → *Remove All Breakpoints* from the menu bar or double-click on the breakpoint symbol in the editor pane. Again, breakpoints can only be removed when the processor is not running.

***Terminate the debug session.***

► Click the red Terminate button to terminate the debug session and return to the CCS Edit perspective.

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio 2 - 27*

*Lab 2 Procedure*

**VARS.INI – An Easier Way to Add Variables**

Recall that earlier in the lab you created two variables – a path variable and a build variable. They were the SAME variable set to the SAME path, but used in two different ways – one was for linking files into your project and the other was used for include search paths during the build.

The variables you created earlier were available on a *project* level. So, if you had two projects open in your workspace, the other project would NOT be able to use the variables that you created.

Now, we’ll show you how to add these variables almost automatically to your WORKSPACE so that ANY project in the workspace can use them.

**23. Using vars.ini to set workspace path and build variables.**

First, let’s look at a new file called vars.ini. ► Select *File* → *Open File* and browse to:

C:\TM4C123G\_LaunchPad\_Workshop\vars.ini

► Click Open

You’ll find the single TIVAWARE\_INSTALL variable listed inside the file:

Before we import this file into the workspace, let’s see where these variables are stored.

► Select *Window* → *Preferences*. When the dialogue appears, ► type “*linked*” into the filter field as shown – then click on *Linked Resources*:

This displays all of your WORKSPACE level path variables. We set these variables at the PROJECT level before. We’re now ready to set them at the WORKSPACE level so that all projects in our workspace can use the same variables.

You could simply add the variable here manually, but importing them from vars.ini is simpler and will set BOTH variables at the same time.

► Type “*build*” into the filter area and click on *Build Variables* as shown:

This is where you can set WORKSPACE level build variables. Again, you could just add the variable now manually, but vars.ini will do this for us.

Both the *Linked Resources* and *Build Variables* areas for your workspace were BLANK – containing no workspace variables at all. That’s about to change...

► Click *Cancel*.

*2 - 28 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio*

*Lab 2 Procedure*

Let’s import the file vars.ini and see what happens....

► Select *File* → *Import,* then expand

the CCS category, click on *Build Variables* (as shown):

► Click *Next* and browse to the location of vars.ini: C:\TM4C123G\_LaunchPad\_Workshop\vars.ini

► Click *Open*, then click *Finish*. ► Then select *Window Preferences* and locate your WORKSPACE path variable and your build variable. Did they show up? It should have

imported the variable listed into both the path and build variable areas (as shown):

► Click OK. Minimize Code Composer.

***Using VARS.INI – Conclusion***

Now, ANY project in your workspace (like all the future labs in this workshop) can use these variables without any more importing. They are part of your workspace. Also, if you export a project and hand it to a friend, these workspace variables will NOT be included in the project. That’s pretty handy. Why? Your friend may have a DIFFERENT install location for the tools. So, if they use the same WORKSPACE VARIABLE names, but different paths, their builds will work just fine. You now have a completely and totally PORTABLE PROJECT.

**Note:** If you change workspaces, you will have to re-import vars.ini to set these

variables again. If your tools installation changes, you’ll have to edit vars.ini and re-import. So be careful.

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio 2 - 29*

*LM Flash Programmer*

**LM Flash Programmer**

LM Flash Programmer is a standalone programming GUI that allows you to program the flash of a Tiva C Series device through multiple ports. Creating the files required for this is a separate build step in Code Composer that it shown on the next page. If you have not done so already, install the LM Flash Programmer onto your PC.

Make sure that Code Composer Studio is not actively running code in the CCS Debug perspective... otherwise CCS and the Flash Programmer may conflict for control of the USB port.

**24. Open LM Flash Programmer**

There should be a shortcut to the LM Flash Programmer on your desktop, double-click it to open the tool. If the shortcut does not appear, go to *Start* → *All Programs* → *Texas Instruments* → *Stellaris* → *LM Flash Programmer* and click on *LM Flash Programmer*.

Your evaluation board should currently be programmed with the lab2 application and it should be running. If the User LED isn’t blinking, press the RESET button on the board. We’re going to program the original application back into the TM4C123GH6PM flash memory.

► Click the Configuration tab. Select the *TM4C123G LaunchPad* from the Quick Set pull-down menu under the Configuration tab. **If *TM4C123G LaunchPad* does not appear, select *LM4F120 LaunchPad* from the list.**

See the user’s guide for information on how to manually configure the tool for targets that are not evaluation boards.

*2 - 30 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio*

*LM Flash Programmer*

**25. Click the Program Tab, then click the Browse button and navigate to:**

c:\TI\TivaWare\_C\_Series-1.1\examples\boards\ek-tm4c123gxl\ qs-rgb\ccs\Debug\qs-rgb.bin

and ► click Open. You may find that clicking on the symbol rather than the file name is easier to navigate.

qs-rgb is the application that was programmed into the flash memory of the TM4C123GH6PM when you removed it from the box.

Note that there are applications here which have been built with each supported IDE.

► Make sure that the following checkboxes are selected:

**26. Program**

► Click the Program button. You should see the programming and verification status at the bottom of the window. After these steps are complete, the quickstart application should be running on your LaunchPad.

**27. Close the LM Flash Programmer**

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio 2 - 31*

*Optional: Creating a bin File for the Flash Programmer*

**Optional: Creating a bin File for the Flash Programmer**

If you want to create a .bin file for use by the stand-alone programmer in any of the labs in this workshop or in your own project, use these steps below.

Remember that the project will have to be open before you can change its properties.

**28. Set Post-Build step to call “tiobj2bin” utility**

► In CCS Project Explorer, right-click on your project and select *Properties*. On the left, click Build and then the *Steps* tab. Paste the following commands into the *Post-build steps Command* box. **Note:** The following four “lines” should be entered as a single line in the *Command*

box. To make it easier, we included a text file that you can copy-paste. Navigate to C:\TM4C123G\_LaunchPad\_Workshop\postbuild.txt to find the complete command line.

"${CCS\_INSTALL\_ROOT}/utils/tiobj2bin/tiobj2bin" "${BuildArtifactFileName}" "${BuildArtifactFileBaseName}.bin" "${CG\_TOOL\_ROOT}/bin/armofd" "${CG\_TOOL\_ROOT}/bin/armhex" "${CCS\_INSTALL\_ROOT}/utils/tiobj2bin/mkhex4bin"

**29. Rebuild your project**

This post-build step will run after your project builds and the .bin file will be in the C:\TM4C123G\_LaunchPad\_Workshop\labx\project\debug folder. You can access this .bin in the CCS Project Explorer in your project by expanding the Debug folder.

If you try to re-build and you receive a message “gmake: Nothing to be done for ‘all’.”, this indicates that no files have changed in your project since the last time you built it. You can force the project to build by first right-clicking the project and then select *Clean Project*. Now you should be able to re-build your project which will run the post-build step to create the .bin file.

30. If you opened lab2 to perform these steps, close it now.

You’re done.

*2 - 32 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio*

*Optional: Creating a bin File for the Flash Programmer*

**Hints and Tips**

There are several issues and errors that users commonly run into during the class. Here are a few and their solutions:

1. **Header files can’t be found**

When you create the main.c file and include the header files, CCS doesn’t know the path to those files and will tell you so by placing a question mark left of those lines. After you change the Compiler and Linker options, these question marks should go away and CCS should find the files during the build. If CCS reports that your header files can’t be found, check the following:

a. Under the Project Properties click Resource on the left. Make sure that your

project is located in: C:\TM4C123G\_LaunchPad\_Workshop\labx\project. If you located it in the labx folder it is possible to adjust the Include and File Search paths. If you located the project in the workspace, your best bet is to remake the project.

b. Review the steps above and assure that your path and build variables are set

properly.

**2. Unresolved symbols**

This is usually the result of step 1b above or you are using a copy of the startup\_ccs.c file that includes the ISR name used in the Interrupts lab. You’ll have to remove the extern declaration and change the timer ISR link back to the default.

**3. Frequency out of range**

This usually means that CCS tried to connect to the evaluation board and couldn’t. This can be the result of the USB drivers or a hardware issue:

a. Unplug and re-plug the board from your USB port to refresh the drivers.

b. Open your Device Manager and verify that the drivers are correctly installed.

c. Assure that your emulator cable is connected to the DEBUG microUSB port,

not the DEVICE port, and make sure the PWR SELECT switch is set to the rightmost DEBUG position.

**4. Error loading dll file**

This can happen in Windows7 when attempting to connect to the evaluation board. This is a Win7 driver installation issue and can be resolved by copying the files: FTCJTAG.dll and ftd2xx.dll to:

C:\CCS5.x\ccsv5\ccs\_base\DebugServer\drivers

and

C:\Windows\System32

Download these files from http://www.ti.com/tool/lm\_ftdi\_driver .

*Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio 2 - 33*

*Optional: Creating a bin File for the Flash Programmer*

**5. Program run tools disappear in the Debug perspective**

The tools aren’t part of the perspective, but part of the Debug window. Somehow you closed the window. Click View → Debug from the menu bar or click the Restore button.

**6. CCS doesn’t prompt for a workspace on startup**

You checked the “don’t ask anymore” checkbox. You can switch workspaces by clicking File → Switch workspace ... or you can do the following: In CCS, click Window → Preferences. Now click the + next to General, Startup and Shutdown, and then click Workspaces. Check the “Prompt for workspace on startup” checkbox and click OK.

**7. The windows have changed in the CCS Edit or Debug perspective from the**

**default and you want them back**

On the CCS menu bar, click Window → Reset Perspective ... and then click Yes.

*2 - 34 Getting Started With the Tiva C Series TM4C123G LaunchPad Workshop - Code Composer Studio*

**TivaWareTM, Initialization and GPIO**

**Introduction**

This chapter will introduce you to TivaWare, the initialization of the device and the operation of the GPIO. The lab exercise uses TivaWare API functions to set up the clock, and to configure and write to the GPIO port.

*Getting Started With the Tiva TM4C123G LaunchPad Workshop - Initialization 3 - 1*

**Agenda**

Introduction to ARM® CortexTM-M4F and Peripherals Code Composer Studio **Introduction to TivaWareTM, Initialization and GPIO** Interrupts and the Timers ADC12 Hibernation Module USB Memory and Security Floating-Point BoosterPacks and grLib Synchronous Serial Interface UART μDMA Sensor Hub PWM

TivaWare...

*Chapter Topics*

**Chapter Topics**

**TivaWareTM, Initialization and GPIO ......................................................................................................3-1**

*Chapter Topics .........................................................................................................................................3-2*

*TivaWare ..................................................................................................................................................3-3*

*Clocking ...................................................................................................................................................3-4*

*GPIO ........................................................................................................................................................3-6*

*Lab 3: Initialization and GPIO ................................................................................................................3-9* Objective..............................................................................................................................................3-9 Procedure ...........................................................................................................................................3-10

*3 - 2 Getting Started With the Tiva TM4C123G LaunchPad Workshop - Initialization*

*TivaWare*

**TivaWare**

**TivaWareTM for C Series Features**

**Peripheral Driver Library** ◆ High-level API interface to complete peripheral set ◆ License & royalty free use for TI Cortex-M parts ◆ Available as object library and as source code ◆ **Programmed into the on-chip ROM**

**USB Stacks and Examples**

◆ USB Device and Embedded Host compliant ◆ Device, Host, OTG and Windows-side examples

◆ Free VID/PID sharing program

**Ethernet**

◆ lwip and uip stacks with 1588 PTP modifications

◆ Extensive examples

**Graphics Library**

**Sensor Library**

◆ Graphics primitive and widgets ◆ 153 fonts plus Asian and Cyrillic

◆ Graphics utility tools

◆ An handling interrupt I2C driven transfers I2C master driver for ◆ A set of drivers for I2C connected sensors ◆ A set of routines for common sensor operations ◆ Three Processing layers: Transport, Sensor and **In System Programming Options**

**Tiva Serial Flash Loader** ◆ Small piece of code that allows programming of the flash without the need for a

debugger interface. ◆ All Tiva C Series MCUs ship with the loader in flash ◆ UART or SSI interface option ◆ The LM Flash Programmer interfaces with the serial flash loader ◆ See application note SPMA029

**Tiva Boot Loader** ◆ Preloaded in ROM or can be programmed at the beginning of flash to act as an

application loader ◆ Can also be used as an update mechanism for an application running on a Tiva

microcontroller. ◆ Interface via UART (default), I2C, SSI, Ethernet, USB (DFU H/D) ◆ Included in the Tiva Peripheral Driver Library with full applications examples

Fundamental Clocks...

*Getting Started With the Tiva TM4C123G LaunchPad Workshop - Initialization 3 - 3*

**Extras** ◆ Wireless protocols ◆ IQ math examples ◆ Bootloaders ◆ Windows side applications

ISP Options...

*Clocking*

**Clocking**

**Fundamental Clock Sources**

**Precision Internal Oscillator (PIOSC)**

◆ 16 MHz ± 3% **Main Oscillator (MOSC) using...**

◆ An external single-ended clock source ◆ An external crystal **Internal 30 kHz Oscillator**

◆ 30 kHz ± 50% ◆ Intended for use during Deep-Sleep power-saving modes **Hibernation Module Clock Source**

◆ 32,768Hz crystal ◆ Intended to provide the system with a real-time clock source

SysClk Sources...

**System (CPU) Clock Sources**

**The CPU can be driven by any of the fundamental clocks ...** ◆ Internal 16 MHz ◆ Main ◆ Internal 30 kHz ◆ External Real-Time **- Plus -** ◆ The internal PLL (400 MHz) ◆ The internal 16MHz oscillator divided by four (4MHz ± 3%)

**Clock Source Drive PLL? Used as SysClk?** Internal 16MHz Yes Yes Internal 16Mhz/4 No Yes Main Oscillator Yes Yes Internal 30 kHz No Yes Hibernation Module No Yes PLL - Yes

Clock Tree...

*3 - 4 Getting Started With the Tiva TM4C123G LaunchPad Workshop - Initialization*

*Clocking*

*Getting Started With the Tiva TM4C123G LaunchPad Workshop - Initialization 3 - 5*

**driverLib API SysCtlClockSet() selects:**

◆ SYSDIV divider setting ◆ OSC or PLL ◆ Main or Internal oscillator ◆ Crystal frequency GPIO...

**Tiva C Series Clock Tree**

*GPIO*

**GPIO**

**General Purpose IO**

◆ **Any GPIO can be an interrupt:**

◆ Edge-triggered on rising, falling or both ◆ Level-sensitive on high or low values ◆ **Can directly initiate an ADC sample sequence or μDMA transfer** ◆ **Toggle rate up to the CPU clock speed on the Advanced High-Performance Bus. 1⁄2 CPU clock speed on the Standard.** ◆ **5V tolerant in input configuration** ◆ **Programmable Drive Strength (2, 4, 8mA or 8mA with slew rate**

**control)** ◆ **Programmable weak pull-up, pull-down, and open drain** ◆ **Pin state can be retained during Hibernation mode**

Pin Mux Utility...

**Pin Mux Utility**

◆ Allows the user to graphically configure the device pin-out ◆ Generates source and header files for use with any of the supported IDE’s

**http://www.ti.com/tool/tm4c\_pinmux**

Masking...

http://www.ti.com/tool/tm4c\_pinmux

*3 - 6 Getting Started With the Tiva TM4C123G LaunchPad Workshop - Initialization*

*GPIO*

**GPIO Address Masking**

**Each address bit-mask each GPIO GPIO and to port port indicate all eight to has 256 a which pins base addresses. are bits address. modified. are Bits to You be 9:2 If modified. can you of the write want address This an to 8-bit modify is bus done value are specific in directly used hardware bits, as to the you this by bit mapping can base mask. use a** The register we Current want to contents change of is the GPIO register Port D is: (0x4005.8000) **00011101**

0x4005.8098. Instead of writing Bits for 9:2 the to (shown GPIO value you Port here) write.

D become directly, write a bit-mask to **00111011**

**GPIOPinWrite(GPIO\_PORTD\_BASE, GPIO\_PIN\_5|GPIO\_PIN\_2|GPIO\_PIN\_1, 0xEB);**

The GIPOPinWrite() Note: you specify function base determines address, bit the mask, correct and address value to for write. the mask. GPIOLOCK ...

The masking technique used on Tiva C Series GPIO is similar to the “bit-banding” technique used in memory. To aid in the efficiency of software, the GPIO ports allow for the modification of individual bits in the **GPIO Data (GPIODATA)** register by using bits [9:2] of the address bus as a mask. In this manner, software can modify individual GPIO pins in a single, atomic read- modify-write (RMW) instruction without affecting the state of the other pins. This method is more efficient than the conventional method of performing a RMW operation to set or clear an individual GPIO pin. To implement this feature, the **GPIODATA** register covers 256 locations in the memory map.

.

*Getting Started With the Tiva TM4C123G LaunchPad Workshop - Initialization 3 - 7*

**GPIO Port D (0x4005.8000)**

The value we will write is 0xEB: **Write Value (0xEB)**

**11101011**

**... 0000010011000** Only the bits marked changed. as “1” in the bit-mask are that New only value the in red GPIO bits Port were D written) (note

*GPIO*

**Critical Function GPIO Protection**

◆ **Six pins on the device are protected against accidental**

**programming:**

• PC3,2,1 & 0: JTAG/SWD

• PD7 & PF0: NMI ◆ **Any write to the following registers for these pins will not be stored unless the GPIOLOCK register has been unlocked:**

• GPIO Alternate Function Select register

• GPIO Pull Up or Pull Down select registers

• GPIO Digital Enable register ◆ **The following sequence will unlock the GPIOLOCK register for**

**PF0 using direct register programming:**

**HWREG(GPIO\_PORTF\_BASE + GPIO\_O\_LOCK) = GPIO\_LOCK\_KEY; HWREG(GPIO\_PORTF\_BASE + GPIO\_O\_CR) |= 0x01; HWREG(GPIO\_PORTF\_BASE + GPIO\_O\_LOCK) = 0;**

◆ **Reading the GPIOLOCK register returns it to lock status**

Lab...

*3 - 8 Getting Started With the Tiva TM4C123G LaunchPad Workshop - Initialization*

*Lab 3: Initialization and GPIO*

**Lab 3: Initialization and GPIO**

**Objective**

In this lab we’ll learn how to initialize the clock system and the GPIO peripheral using TivaWare. We’ll then use the GPIO output to blink an LED on the evaluation board.

**Lab 3: Initialization and GPIO**

◆ Configure the system clock ◆ Enable and configure GPIO ◆ Use on the a software evaluation delay board

to toggle an LED Agenda ...

*Getting Started With the Tiva TM4C123G LaunchPad Workshop - Initialization 3 - 9*

USB Emulation Connection

*Lab 3: Initialization and GPIO*

**Procedure**

***Create lab3 Project***

1. ► Maximize Code Composer. On the CCS menu bar select File → New → CCS Project.

Make the selections shown below. Make sure to **uncheck** the “Use default location” checkbox and select the correct path to the project folder as shown. In the variant box, just type “123G” to narrow the results in the right-hand box. In the Project templates and examples window, select **Empty Project (with main.c)**. Click Finish.

When the wizard completes, click the next to lab3 in the Project Explorer pane to expand the project. Note that Code Composer has automatically added a mostly empty main.c file to your project as well as the startup file.

Note: We placed a file called main.txt in the lab3/project folder which contains the final code for the lab. If you run into trouble, you can refer to this file.

*3 - 10 Getting Started With the Tiva TM4C123G LaunchPad Workshop - Initialization*

*Lab 3: Initialization and GPIO*

***Header Files***

2. ► Delete the current contents of main.c.

TivaWareTM is written using the ISO/IEC 9899:1999 (or C99) C programming standards along with the Hungarian standard for naming variables. The C99 C programming conventions make better use of available hardware, including the IEE754 floating point unit. To keep everything looking the same, we’re going to use those guidelines.

► Type (or cut/paste from the pdf file) the following lines into main.c to include the header files needed to access the TivaWare APIs as well as a variable definition:

**#include** <stdint.h> **#include** <stdbool.h> **#include** "inc/hw\_memmap.h" **#include** "inc/hw\_types.h" **#include** "driverlib/sysctl.h" **#include** "driverlib/gpio.h"

uint8\_t ui8PinData=2;

The use of the < > restricts the search path to only the specified path. Using the " " causes the search to start in the project directory. For includes like the two standard ones, you want to assure that you’re accessing the original, standard files ... not one’s that may have been modified.

**stdint.h: V**ariable definitions for the C99 standard

**stdbool.int:** Boolean definitions for the C99 standard

**hw\_memmap.h** : Macros defining the memory map of the Tiva C Series device. This includes defines such as peripheral base address locations such as GPIO\_PORTF\_BASE.

**hw\_types.h** : Defines common types and macros

**sysctl.h** : Defines and macros for System Control API of DriverLib. This includes API functions such as SysCtlClockSet and SysCtlClockGet.

**gpio.h** : Defines and macros for GPIO API of DriverLib. This includes API functions such as GPIOPinTypePWM and GPIOPinWrite.

**uint8\_t ui8PinData=2**; : Creates an integer variable called ui8PinData and initializes it to 2. This will be used to cycle through the three LEDs, lighting them one at a time. Note that the C99 type is an 8-bit unsigned integer and that the variable name reflects this.

You will see question marks to the left of the include lines in main.c displayed in the edit pane, telling us that the include files can’t be found. We’ll fix this later.

*Getting Started With the Tiva TM4C123G LaunchPad Workshop - Initialization 3 - 11*

*Lab 3: Initialization and GPIO*

***main() Function***

3. Let’s drop in a template for our main function.

► Leave a line for spacing and add this code after the previous declarations:

**int main(void)**

**{ }** If you type this in, notice that the editor will automatically add the closing brace when you add the opening one. Why wasn’t this thought of sooner?

*3 - 12 Getting Started With the Tiva TM4C123G LaunchPad Workshop - Initialization*

*Lab 3: Initialization and GPIO*

***Clock Setup***

4. Configure the system clock to run using a 16MHz crystal on the main oscillator, driving the 400MHz PLL. The 400MHz PLL oscillates at only that frequency, but can be driven by crystals or oscillators running between 5 and 25MHz. There is a default /2 divider in the clock path and we are specifying another /5, which totals 10. That means the System Clock will be 40MHz.

► Enter this single line of code inside main():

**SysCtlClockSet(SYSCTL\_SYSDIV\_5|SYSCTL\_USE\_PLL|SYSCTL\_XTAL\_16MHZ|SYSCTL\_OSC\_MAIN);**

The diagram below is an abbreviated drawing of the clock tree to emphasize the System Clock path and choices. Note the darkened path.

The diagram below is an excerpt from the LaunchPad board schematic. Note that the crystal attached to the main oscillator inputs is 16MHz, while the crystal attached to the real-time clock (RTC) inputs is 32,768Hz.

*Getting Started With the Tiva TM4C123G LaunchPad Workshop - Initialization 3 - 13*

*Lab 3: Initialization and GPIO*

***GPIO Configuration***

5. Before calling any peripheral specific driverLib function, we must enable the clock

for that peripheral. If you fail to do this, it will result in a Fault ISR (address fault).This is a common mistake for new Tiva C Series users. The second statement below configures the three GPIO pins connected to the LEDs as outputs. The excerpt below of the LaunchPad board schematic shows GPIO pins PF1, PF2 and PF3 are connected to the LEDs.

► Leave a line for spacing, then enter these two lines of code inside main() after the line in the previous step.

**SysCtlPeripheralEnable(SYSCTL\_PERIPH\_GPIOF); GPIOPinTypeGPIOOutput(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3);**

The base addresses of the GPIO ports listed in the User Guide are shown below. Note that they are all within the memory map’s peripheral section shown in module 1. APB refers to the Advanced Peripheral Bus, while AHB refers to the Advanced High- Performance Bus. The AHB offers better back-to-back performance than the APB bus. GPIO ports accessed through the AHB can toggle every clock cycle vs. once every two cycles for ports on the APB. In power sensitive applications, the APB would be a better choice than the AHB. In our labs, GPIO\_PORTF\_BASE is 0x40025000.

**GPIO Port A (APB): 0x4000.4000 GPIO Port A (AHB): 0x4005.8000 GPIO Port B (APB): 0x4000.5000 GPIO Port B (AHB): 0x4005.9000 GPIO Port C (APB): 0x4000.6000 GPIO Port C (AHB): 0x4005.A000 GPIO Port D (APB): 0x4000.7000 GPIO Port D (AHB): 0x4005.B000 GPIO Port E (APB): 0x4002.4000 GPIO Port E (AHB): 0x4005.C000 GPIO Port F (APB): 0x4002.5000 GPIO Port F (AHB): 0x4005.D000**

*3 - 14 Getting Started With the Tiva TM4C123G LaunchPad Workshop - Initialization*

*Lab 3: Initialization and GPIO*

***while() Loop***

6. Finally, create a while(1) loop to send a “1” and “0” to the selected GPIO pin, with an

equal delay between the two.

**SysCtlDelay()** is a loop timer provided in TivaWare. The count parameter is the loop count, not the actual delay in clock cycles. Each loop is 3 CPU cycles.

To write to the GPIO pin, use the GPIO API function call GPIOPinWrite. Make sure to read and understand how the GPIOPinWrite function is used in the datasheet. The third data argument is not simply a 1 or 0, but represents the entire 8-bit data port. The second argument is a bit-packed mask of the data being written.

In our example below, we are writing the value in the ui8PinData variable to all three GPIO pins that are connected to the LEDs. Only those three pins will be written to based on the bit mask specified. The final instruction cycles through the LEDs by making ui8PinData equal to 2, 4, 8, 2, 4, 8 and so on. Note that the values sent to the pins match their positions; a “one” in the bit two position can only reach the bit two pin on the port.

Now might be a good time to look at the Datasheet for your Tiva C Series device. Check out the GPIO chapter to understand the unique way the GPIO data register is designed and the advantages of this approach.

► Leave a line for spacing, and then add this code after the code in the previous step.

**while(1) { GPIOPinWrite(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3, ui8PinData); SysCtlDelay(2000000); GPIOPinWrite(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3, 0x00); SysCtlDelay(2000000); if(ui8PinData==8) {ui8PinData=2;} else {ui8PinData=ui8PinData\*2;} }** If you find that the indentation of your code doesn’t look quite right, ► select all of your code by clicking CTRL-A and then right-click on the selected code. Select **Source** → **Correct Indentation**. Notice the other great stuff under the **Source** and **Surround With** selections.

*Getting Started With the Tiva TM4C123G LaunchPad Workshop - Initialization 3 - 15*

*Lab 3: Initialization and GPIO*

7. ► Click the Save button to save your work. Your code should look something like this:

**#include** <stdint.h> **#include** <stdbool.h> **#include** "inc/hw\_memmap.h" **#include** "inc/hw\_types.h" **#include** "driverlib/sysctl.h" **#include** "driverlib/gpio.h"

uint8\_t ui8PinData=2;

**int main**(**void**) {

**SysCtlClockSet**(SYSCTL\_SYSDIV\_5|SYSCTL\_USE\_PLL|SYSCTL\_XTAL\_16MHZ|SYSCTL\_OSC\_MAIN);

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_GPIOF); **GPIOPinTypeGPIOOutput**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3);

**while**(1) {

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1| GPIO\_PIN\_2| GPIO\_PIN\_3, ui8PinData); **SysCtlDelay**(2000000); **GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1|GPIO\_PIN\_2|GPIO\_PIN\_3, 0x00); **SysCtlDelay**(2000000); **if**(ui8PinData==8) {ui8PinData=2;} **else** {ui8PinData=ui8PinData\*2;} } }

If you’re having problems, you can cut/paste this code into main.c or you can cut/paste from the main.txt file in your Project Explorer pane.

If you were to try building this code now (please don’t), it would fail. Note the question marks next to the include statements ... CCS has no idea where those files are located ... we still need to set our build options.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**NOTE**: There is a delay of 3 to 6 clock cycles between enabling a peripheral and being able to use it. In most cases, the amount of time required by the API coding itself prevents any issues, but there are situations where you may be able to cause a system fault by attempting to access the peripheral before it becomes available.

A good programming habit is to interleave your peripheral enable statements as follows:

**Enable ADC Enable GPIO Config ADC Config GPIO**

This interleaving will prevent any possible system faults without incorporating software delays.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*3 - 16 Getting Started With the Tiva TM4C123G LaunchPad Workshop - Initialization*

*Lab 3: Initialization and GPIO*

***Startup Code***

8. In addition to the main file you have created, you will also need a startup file specific to

the tool chain you are using. This file contains the vector table, startup routines to copy initialized data to RAM and clear the bss section, and default fault ISRs. The New Project wizard automatically added a copy of this file into the project for us.

► Double-click on tm4c123gh6pm\_startup\_ccs.c in your Project Explorer pane and take a look around. Don’t make any changes at this time. Close the file.

***Set the Build Options***

9. ► Right-click on Lab3 in the Project Explorer pane and select Properties. Click **Include Options** under **ARM Compiler**. In the bottom, **include search path** pane, click the Add button and add the following search path:

**${TIVAWARE\_INSTALL}**

Remember that those are braces, not parentheses. This is the path we created earlier by using the vars.ini file in the lab2 project. Since those paths are defined at the workspace level, we can simply use it again here.

► Click OK.

After a moment, CCS will refresh the project and you should see the question marks dis- appear from the include lines in main.c.

*Getting Started With the Tiva TM4C123G LaunchPad Workshop - Initialization 3 - 17*

*Lab 3: Initialization and GPIO*

**10. Add the Driver Library File**

The driverlib.lib file needs to be in the lab3 project. In lab2 we added a link to this file. You can see it under your lab2 project in the Project Explorer pane. Can it be as simple as dragging it over? Let’s try it.

► Click and hold driverlib.lib under the lab2 project in the Project Explorer pane. ► Drag it onto the lab3 project and release7. You should now see the file under lab3.

The file that was linked to lab2 is now linked to lab3. That was even easier.

11. It can be easy to get confused and mistakenly build or work on the wrong project or file. To reduce that possibility, ► right-click on lab2 and select *Close Project*. This will collapse the project and close any open files you have from the project. You can open it again at any time. ► Click on the lab3 project name to make sure the project is active. It will say **lab3 [Active – Debug]**. This tells you that the lab3 project is active and that the build configuration is debug.

**12. Stack Considerations**

► Test build the lab3 to check for errors by clicking the Build (Hammer)

button. Note that a warning appears in the Problems pane in the lower right of CCS. This error; “creating .stack section with default size of 0x800...” tells us that the stack size was not specified. We can eliminate this warning by specifying the stack size(s).

► Right-click on the lab3 project in the Project Explorer pane and select Properties. Expand *Build* → *ARM Linker* and click on *Basic Options*. Find the *Heap size* and *Set C system stack size* boxes as shown below.

► Enter *0* for the *Heap size* and *100* for the *C system stack size* and click OK. We won’t be

using the heap in these labs and our need for a C stack is very limited. Failure to monitor the size of your stack(s) can result in significant amount of memory being wasted. Test build again to make sure the warning no longer appears.

These settings will be made for you in the rest of the labs.

*3 - 18 Getting Started With the Tiva TM4C123G LaunchPad Workshop - Initialization*