Digital Signal Processing and Applications with the OMAP-L138 eXperimenter

DONALD REAY

WILEY

Digital Signal Processing and Applications with the OMAP-L138 eXperimenter

Digital Signal Processing and Applications with the OMAP-L138 eXperimenter

Donald Reay
Heriot-Watt University

WILEY
A JOHN WILEY & SONS, INC., PUBLICATION

Published by John Wiley & Sons, Inc., Hoboken, New Jersey

Published simultaneously in Canada

Library of Congress Cataloging-in-Publication Data:

Reay, Donald (Donald S.)
Digital signal processing and applications with the OMAP-L138 eXperimenter / Donald Reay.
p. cm.
Includes bibliographical references and index.
ISBN 978-0-470-93686-3 (hardback)
1. Signal processing–Digital techniques–Experiments. 2. Microprocessors–Experiments. I. Title.
TK5102.9.R4325 2012
621.382’2078–dc23

2011038412

Printed in the United States of America
10 9 8 7 6 5 4 3 2 1

To Reiko

Contents

Preface

This book continues the series started in 1990 by Rulph Chassaing and Darrell Horning's *Digital Signal Processing with the TMS320C25* and which has reflected the development of successive generations of digital signal processors by Texas Instruments. More specifically, each book in this series has complemented a different one of the inexpensive DSP development tools promoted by the Texas Instruments University Programme for teaching purposes. A consistent theme in the books has been the provision of a large number of simple example programs illustrating DSP concepts in real-time in a laboratory setting.

It was Rulph Chassaing's belief, and also mine, that hands-on teaching of DSP, using hardware development kits and laboratory test equipment to process analog audio frequency signals, is a valuable and effective way of reinforcing the theory taught in lectures.

The contents of the books, insofar as they concern fundamental concepts of digital signal processing such as analog-to-digital and digital-to-analog conversion, FIR and IIR filtering, the Fourier transform, and adaptive filtering, have changed little. Every year, in the context of university teaching, brings another set of students wanting to study this material. However, each successive book has concerned a different hardware development kit. The latest hardware development kit to be promoted by the Texas Instruments University Programme is the Logic PD OMAP-L138 eXperimenter.

This book is suitable for senior undergraduate and postgraduate electrical engineering students who have a basic knowledge of C programming and of linear systems theory.

The architecture of Texas Instruments' DSP devices has reached a level of complexity that places assembly language programming out of reach of such students. Certainly, I have found that it is beyond the scope and time available in a digital signal processing class. Even some of the optimized DSP functions supplied by Texas Instruments in support libraries are written in C rather than assembly language.

For this reason, this book does not contain chapters concerning processor architecture or assembly language programming.

The OMAP-L138 is a dual-core processor, the capabilities of which are far beyond what can be covered in a single text. This book uses only a fraction of its features in order to provide teaching materials specifically for DSP.

It is intended and hoped that this book will prove a useful resource for anyone involved in teaching or learning DSP and as a starting point for teaching or learning more.

I am grateful to Robert Owen and Cathy Wicks at the TI University Programme for their support and encouragement, to George Telecki at Wiley for his patience, to Keith Brown at Heriot-Watt University for his help in testing some of the example programs and for his suggestions, and to He Xiangning at Zhejiang University for giving me the opportunity to try out the example programs in a teaching environment with some very able students. But above all, I thank Rulph Chassaing for inspiring me to get involved in teaching hands-on DSP.

The ftp site (ftp://ftp.wiley.com/public/sci_tech_med/signal_processing) contains the code for the example programs described in the book.

DONALD REAY

List of Examples

Chapter 1

OMAP-L138 Development System

- OMAP-L138 processor
- Code Composer Studio™ IDE version 4
- Use of the OMAP-L138 eXperimenter
- Programming examples

This chapter gives an overview of the OMAP-L138 processor and Logic PD's Zoom OMAP-L138 eXperimenter development system. It describes how to install and start using version 4 of Texas Instruments (TI) Code Composer Studio integrated development environment (IDE). Two example programs that demonstrate hardware and software features of the eXperimenter board and of the Code Composer Studio IDE are presented. It is recommended strongly that you review these examples before proceeding to subsequent chapters.

1.1 INTRODUCTION

The Logic PD Zoom OMAP-L138 eXperimenter kit is a low-cost development platform for the Texas Instruments OMAP-L138 processor. This device is a dual-core system on a chip comprising an ARM926EJ-S general-purpose processor (GPP) and a TMS320C6748 digital signal processor. In addition, a number of peripherals and interfaces are built into the OMAP-L138 as shown in Figure 1.1.

The eXperimenter makes a significant number of the OMAP-L138 interfaces available to the user, as shown in Figure 1.2. This book is concerned with the development of real-time digital signal processing (DSP) applications and therefore makes use of the DSP (C6748) side of the device and of the TLC320AIC3106 (AIC3106) analog interface circuit (codec) connected to the OMAP-L138's

Digital Signal Processing and Applications with the OMAP-L138 eXperimenter, Donald Reay.

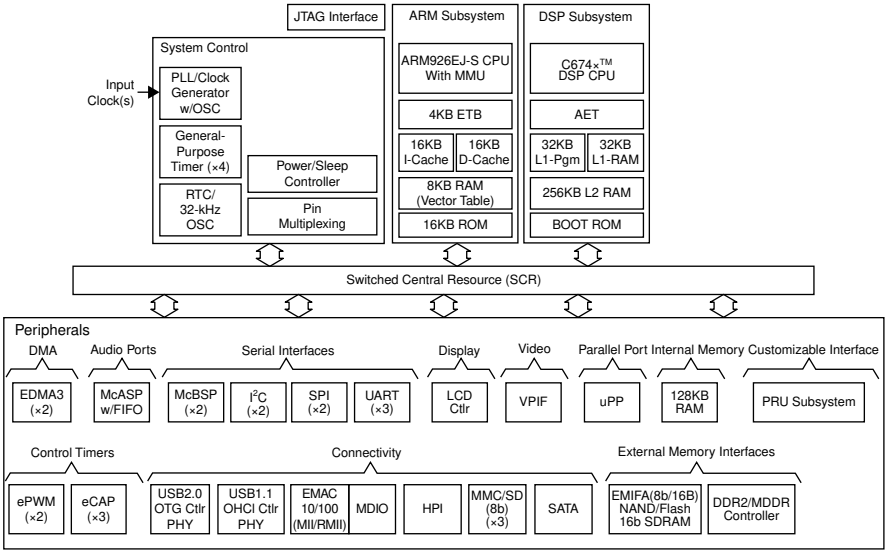

Figure 1.1 Functional block diagram of OMAP-L138 processor. (courtesy of Texas Instruments)

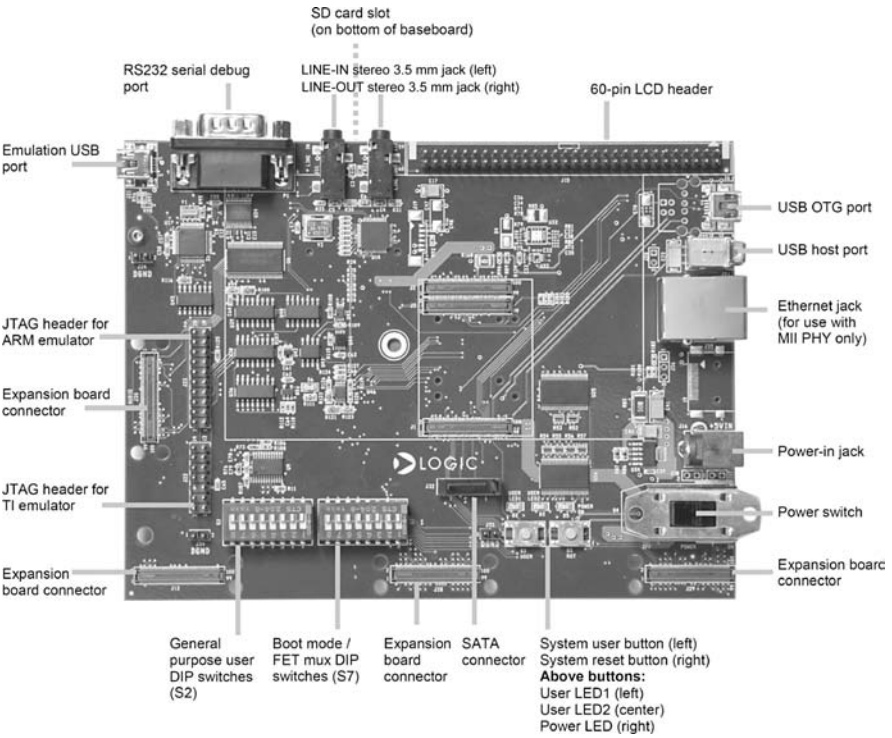

Figure 1.2 Logic PD Zoom OMAP-L138 eXperimenter baseboard (courtesy of Logic PD).

multichannel audio serial port (McASP). The ARM side of the device is not used by the examples in this book. Connection to a host PC running the Code Composer Studio IDE is via XDS100v1 JTAG emulation built in to the eXperimenter. The Code Composer Studio IDE enables software written in C or assembly language to be compiled and/or assembled, linked, and downloaded to run on the C6748. Details of the OMAP-L138, TMS320C6748, TLC320AIC3106, eXperimenter, and Code Composer Studio IDE can be found in their associated datasheets [1–5] and in the TI wiki [6]. The purpose of this chapter is to introduce the installation and use of the eXperimenter for hands-on DSP experiments.

1.1.1 Digital Signal Processors

A digital signal processor is a specialized form of microprocessor. Its architecture and instruction set are optimized for real-time digital signal processing. Typical optimizations include hardware multiply accumulate (MAC) provision, hardware circular and bit-reversed addressing capabilities (for efficient implementation of data buffers and fast Fourier transform (FFT) computation), and Harvard architecture (independent program and data memory systems). In many respects, digital signal processors resemble microcontrollers. Typically, they provide single-chip computer solutions integrating on-board volatile and nonvolatile memory and a range of peripheral interfaces, and have a small footprint, making them ideal for embedded applications. In addition, digital signal processors tend to have low power consumption requirements. This attribute has been extremely important in establishing the use of digital signal processors in cellular handsets. However, the distinctions between digital signal processors and other more general-purpose microprocessors are blurred. No strict definition of a digital signal processor exists and semiconductor manufacturers apply the term to products exhibiting some, but not necessarily all, of the above characteristics as they see fit.

Digital signal processors are used for a wide range of applications, from communications and control to speech and image processing. They are found in cellular phones, disk drives, radios, printers, MP3 players, HDTV, digital cameras, and so on. Specialized (particularly in terms of their on-board peripherals) DSPs are used in electric motor drives and in a range of associated automotive and industrial applications. Overall, digital signal processors are concerned primarily with real-time signal processing. Real-time processing means that the processing must keep pace with some external event, whereas non real-time processing has no such timing constraint. The external event to keep pace with is usually the analog input. While analog-based systems with discrete electronic components including resistors and capacitors are sensitive to temperature changes, DSP-based systems are less affected by environmental conditions such as temperature. Digital signal processors embody the major advantages of microprocessors. They are easy to use, flexible, and economical.

Texas Instruments OMAP-L138 device combines a C6748 DSP with an ARM926EJ-S general-purpose processor to produce a dual-core solution for

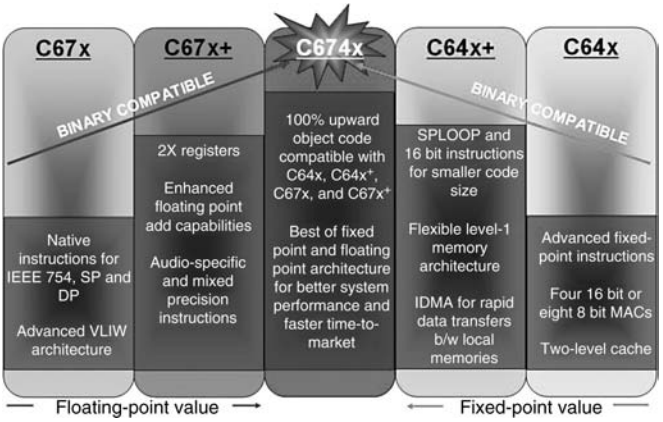

Figure 1.3 The C674x combines C64x fixed-point and C67x floating-point DSP architectures. (courtesy of Texas Instruments).

handheld and other embedded applications. ARM926EJ-S provides the benefits of a 32-bit RISC processor, well suited to implementing user interfaces and running operating systems.

C6748 is a member of the Texas Instruments TMS320C6000™ DSP family of digital signal processors. Its architecture is very well suited to numerically intensive calculations and it is one of TI's most powerful digital signal processors. More specifically, as a member of the C674x family, it combines C64x™ DSP fixed-point and C67x™ DSP floating-point architectures in one core, as illustrated in Figure 1.3.

1.2 HARDWARE AND SOFTWARE TOOLS

Most of the examples presented in this book involve the development and testing of short programs intended to demonstrate fundamental DSP concepts in a laboratory setting. To perform the experiments described in the book, a number of hardware and software tools are required.

(1) ***A Logic PD Zoom OMAP-L138 eXperimenter kit*** This package includes the following:
   (a) Three separate circuit boards, a baseboard, a 4.3″ LCD, and an OMAP-L138 SOM-M1. OMAP-L138 SOM-M1 must be connected to the baseboard, as described in the instructions included in the eXperimenter kit. The LCD is not used in the experiments in this book and therefore need not be connected to the baseboard.
   (b) A universal serial bus (USB) cable that connects the eXperimenter board to a host PC.
   (c) A 5 V universal power supply for the eXperimenter board.

(2) ***A host PC*** This is used to run the Code Composer Studio IDE. The eXperimenter board is connected to a USB port on the host PC.

(3) ***Code Composer Studio software, version 4*** The eXperimenter kit includes a DVD containing Code Composer Studio software version 4. Alternatively, the Code Composer Studio IDE may be downloaded from the Texas Instruments wiki at *http://processors.wiki.ti.com/index.php/Download_CCS*. Code Composer Studio software provides an IDE, bringing together C compiler, assembler, linker, and debugger.

(4) ***Board support library*** Board-level support routines specific to eXperimenter are not supplied with the kit, but may be downloaded from the Logic PD website at *http://www.logicpd.com/product-support*. Access to the board support libraryBSL file `1017292A_OMAP-L138_GEL_BSL_Files_v2.3.zip` requires a login and product registration.

(5) ***TMS320C674x DSP library*** A number of example programs make use of optimized DSP routines from this library. It can be downloaded from the Texas Instruments wiki at *http://processors.wiki.ti.com/index.php/C674x_DSPLIB* or from the Texas Instruments website at *http://focus.ti.com/docs/toolsw/folders/print/sprc900.html*.

(6) ***DSP/BIOS™ software Kernel Foundation Platform Support Package (PSP)*** This is used for examples in Chapter 7. Details of how to download file `BIOSPSP_01_30_00_06_Setup.exe`, which is part of the OMAP-L138 and C6748 software development kits (SDKs) may be found at *http://processors.wiki.ti.com/index.php/GSG_C6748:_Installing_the_SDK_Software*.

(7) ***An oscilloscope, spectrum analyzer, signal generator, headphones, microphone, and loudspeakers*** The experiments presented in subsequent chapters of this book are intended to demonstrate digital signal processing concepts in real-time, using audio frequency analog input and output signals. In order to appreciate these concepts and to get the greatest benefit from the experiments, some forms of signal source and sink are required. As a bare minimum, an audio source with line level output and either headphones or loudspeakers are required. Greater benefit will be accrued if a signal generator is used to generate sinusoidal, and other, test signals and if an oscilloscope and spectrum analyzer are used to display, measure, and analyze input and output signals. Many modern digital oscilloscopes incorporate FFT functions, allowing the frequency content of signals to be displayed. Alternatively, a number of software packages that use a PC equipped with a sound card to implement virtual instruments are available. In this book, *Goldwave* is used, which may be downloaded from *www.goldwave.com*.

(8) The files and example programs listed and discussed in this book are included on the partner website *ftp://ftp.wiley.com/public/sci_tech_med/signal_processing*.

1.2.1 Zoom OMAP-L138 eXperimenter Board

The eXperimenter board is a powerful, yet inexpensive, development system with the necessary hardware and software support tools for real-time signal processing [4]. From the point of view of the example programs in this book, it is a complete DSP system. The board, which measures approximately 5×7 inches, includes a 375 MHz OMAP-L138 processor and a 16-bit stereo codec TLV320AIC3106 (AIC3106) for analog input and output. Numerous other interfaces are provided by the eXperimenter but are not used by the example programs in this book.

The onboard codec AIC3106 [3] uses sigma–delta technology that provides analog-to-digital conversion (ADC) and digital-to-analog conversion (DAC) functions. It uses a 24.576 MHz system clock and its sampling rate can be selected from a range of alternative settings from 8 to 48 kHz.

The eXperimenter boards each include 128 MB of synchronous dynamic RAM (mDDR SDRAM) and 8 MB of NOR flash memory. Two 3.5 mm jack socket connectors on the boards provide analog input and output: LINE IN for line level input and LINE OUT for line level output. The status of eight user DIP switches on the board can be read from within a program running on the processor and provide a simple means of user interaction. The states of two LEDs on the board can be controlled from within a program running on the processor.

1.2.2 C6748 Processor

The DSP core in the OMAP-L138 device (L138) is a C6748 DSP based on Texas Instruments very long instruction word (VLIW) architecture and is very well suited to numerically intensive algorithms. The internal program memory is structured so that a total of eight instructions can be fetched every cycle. With a clock rate of 375 MHz, the C6748 is capable of fetching eight 32-bit instructions every 1/(375 MHz) or 2.67 ns. As part of the C674x family, it incorporates both floating-point and fixed-point architectures in one core.

Features of the C6748 include 326 kB of internal memory (32 kB of L1P program RAM/cache, 32 kB of L1D data RAM/cache, and 256 kB of L2 RAM/cache), eight functional or execution units composed of six ALUs and two multiplier units, an external memory interface addressing 256 MB of 16-bit mDDR SDRAM, and 64 32-bit general-purpose registers. In addition, the OMAP-L138 features 128 kB of on-chip RAM shared by its C6748 and ARM9 processor cores [1].

1.2.3 Code Composer Studio IDE

Code Composer Studio software provides an IDE for real-time digital signal processing applications based on the C programming language. It incorporates a C compiler, an assembler, and a linker. It has graphical capabilities and supports real-time debugging. Version 4 of the Code Composer Studio IDE is based on the

open-source Eclipse framework [7], widely used in embedded systems development.

Code Composer Studio software is project based. A Code Composer Studio software project comprises all the files (or links to all the files) required in order to generate an executable file. In addition, a Code Composer Studio software project contains information about exactly how files are to be used in order to generate an executable file. Compiler/linker options can be specified.

A number of debugging features are available in Code Composer Studio software, including setting breakpoints and watching variables, viewing memory, registers, and mixed C and assembly code, graphing results, and monitoring execution time.

Communication between the Code Composer Studio IDE and the eXperimenter is via a USB connection and the XDS100v1 JTAG emulation [8] built into the board. Compared to previous versions, Code Composer Studio software version 4 separates code development and debugging activities through the use of perspectives. Perspectives are sets of windows, views, and menus and as a default Code Composer Studio software provides a default *C/C++* perspective, including, among others, *Project View*, *Editor*, and *Outline* windows, and a default *Debug* perspective, including *Debug*, *Disassembly*, and *Console* views. Users may customize these perspectives or create new ones. The *View* menu gives an idea of the many other windows available.

1.2.4 Installation of Code Composer Studio Software Version 4 and Support Files

The example programs described in this book are intended to be used with Code Composer Studio software version 4 and were tested using version 4.2.1. Code Composer Studio software is supplied on a DVD as part of the eXperimenter kit or alternatively may be downloaded from the Texas Instruments wiki at *http://processors.wiki.ti.com/index.php/Download_CCS*. Installation instructions for Code Composer Studio software are included on the DVD. A typical (default) location for the files is `c:\Program Files\Texas Instruments\ccsv4`, but this is not mandatory. The default location for the Code Composer Studio IDE was used during preparation of the example programs in this book. Once installed, an icon with the label *Code Composer Studio v4* should appear on the desktop, as shown in Figure 1.4.

The example programs make use of the Logic PD BSL which may be downloaded from http://www.logicpd.com/product-support. During preparation of this book and testing of the example programs, the BSL was installed at `c:\omapl138`.

Some example programs make use of the optimized c674x DSPLIB library of digital signal processing routines [9] that may be downloaded from *http://processors.wiki.ti.com/index.php/C674x_DSPLIB*. During preparation of this book and testing of the example programs, it was installed at `c:\C6748_dsp_1_00 _00_11`.

Figure 1.4 Code Composer Studio software desktop icon.

Example programs in Chapter 7 make use of the DSP/BIOS software kernel foundation PSP. During testing of the example programs, the SDK that includes the PSP was installed according to the instructions at *http://processors.wiki.ti.com/index.php/GSG_C6748:_Installing_the_SDK_Software*, resulting in installation of PSP at `c:\C6748_dsp_1_00_00_11\pspdrivers_01_30_01`.

The files accompanying this book should be copied to `c:\eXperimenter` so that, for example, files relating to this chapter may be found in folder `c:\eXperimenter\L138_chapter1`.

Alternative locations for the various software tools described are possible, but corresponding changes to some of the *Build Properties* within Code Composer Studio software that are shown in this book would have to be made. The path names used and detailed instructions given in this book assume that the software tools have been installed as just described.

1.3 INITIAL TEST OF THE EXPERIMENTER USING A PROGRAM SUPPLIED WITH THIS BOOK

Follow these instructions in order to quickly test the correct installation of the software tools and the eXperimenter board:

(1) Connect the eXperimenter board to the host PC using the USB cable provided. There are two mini-USB sockets on the eXperimenter. The socket used for connection to the host PC is *Emulation USB port* (J21), located adjacent to the 9-way D-type RS232 *serial debug port* (Figure 1.2).

(2) Make sure that all the *Boot mode* DIP switches (S7) are OFF, except for DIP switches #5 and #8 that should be ON (Figure 1.2). This sets the appropriate *Boot Mode* (EMU Debug) for the use made of the eXperimenter in this book.

(3) Connect a line level audio source, for example, the output from a PC sound card, to the LINE IN socket on the eXperimenter. Make sure that the output level from the source is sufficiently low that it will not damage the input circuits of the AIC3106 codec.

(4) Connect either headphones or loudspeakers to the LINE OUT socket on the eXperimenter.

(5) Connect the power supply provided to the eXperimenter board.

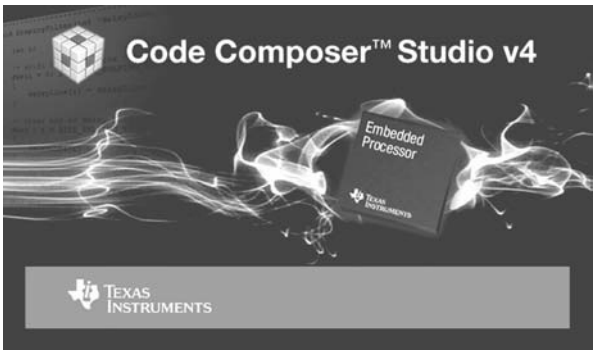

Figure 1.5 Code Composer Studio software version 4 splash screen.

(6) Switch on the eXperimenter using the Power Switch. The *Power On* LED (D5) should light.

(7) Launch Code Composer Studio software by double-clicking on the *Code Composer Studio v4* icon on the desktop. You should see a splash screen as shown in Figure 1.5 and then a pop-up window similar to that shown in Figure 1.6.

(8) Enter `c:\eXperimenter\L138_chapter1` as a workspace, as shown in Figure 1.6 and click *OK*. Next, you should see a welcome screen as shown in Figure 1.7. Click on the *Start using CCS* icon in the top right-hand corner and Code Composer Studio should start in the *C/C++* perspective, but show no projects in the *Project View* window.

The Code Composer Studio software version 4 debugger makes use of a *Target Configuration* file containing details of the hardware system being debugged. For the example programs in this book, target configuration file `L138_experimenter.ccxml` is provided. It is located in folder `c:\eXperimenter\L138_support`.

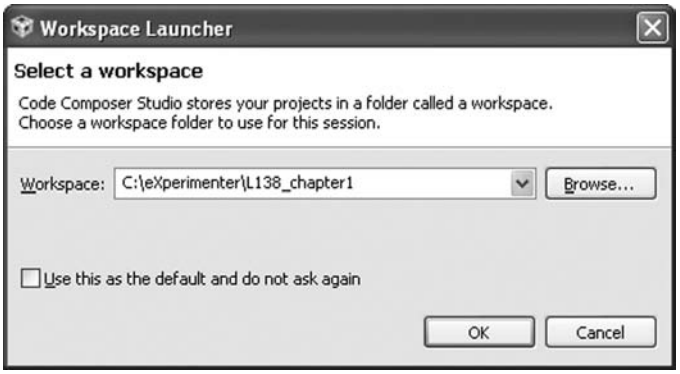

Figure 1.6 Code Composer Studio software version 4 pop-up window.

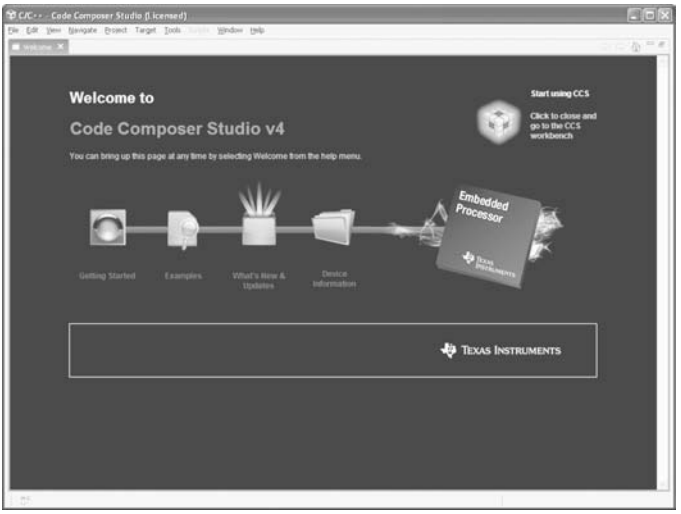

Figure 1.7 Code Composer Studio software version 4 Welcome screen.

(9) Copy the target configuration file `L138_eXperimenter.ccxml` from `c:\eXperimenter\L138_support` to the default location used by the Code Composer Studio IDE for target configuration files, that is, `c:\Documents and Settings\YOUR_ID\user\CCSTargetCon-figurations` if you are using Window XP, or `c:\User\Your_ID\user\CCSTargetConfigurations` if you are using Windows 7.

(10) Copy file `C6748.gel` from folder `c:\eXperimenter\L138_support` to `c:\Documents and Settings\YOUR_ID\user\CCSTargetConfigurations` if you are using Window XP, or `c:\User\Your_ID\user\CCSTargetConfigurations` if you are using Windows 7. This general extension language (GEL) script is run every time you *Connect to Target* in the debugger and carries out a number of important initialization procedures on the eXperimenter.

(11) In the *C/C ++* perspective, select *View* > *Target Configurations*, right-click on `L138_eXperimenter.ccxml`, under *User Defined* and select *Set as Default*. The *Project View* window should then appear as shown in Figure 1.8.

(12) Launch the debugger by selecting *Target* > *Launch TI Debugger*. This should cause Code Composer Studio to switch from the *C/C ++* perspective to the *Debug* perspective, including a *Debug* window as shown in Figure 1.9. If Code Composer Studio software does not automatically switch to the *Debug* perspective, you can do so using the *Debug* button in the top right-hand corner of the Code Composer Studio IDE window, as shown in Figure 1.10.