

Practical Microcontroller Engineering with ARM[®] Technology

IEEE Press
445 Hoes Lane
Piscataway, NJ 08854

IEEE Press Editorial Board
Tariq Samad, *Editor in Chief*

George W. Arnold
Dmitry Goldgof
Ekram Hossain
Mary Lanzerotti

Vladimir Lumelsky
Pui-In Mak
Jeffrey Nanzer
Ray Perez

Linda Shafer
Zidong Wang
MengChu Zhou
George Zobrist

Kenneth Moore, *Director of IEEE Book and Information Services (BIS)*

Practical Microcontroller Engineering with ARM[®] Technology

Ying Bai

*Department of Computer Science and Engineering
Johnson C. Smith University
Charlotte, North Carolina*

 **IEEE**
IEEE PRESS

WILEY

Copyright © 2016 by The Institute of Electrical and Electronics Engineers, Inc.

Published by John Wiley & Sons, Inc., Hoboken, New Jersey. All rights reserved
Published simultaneously in Canada

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning, or otherwise, except as permitted under Section 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923, (978) 750-8400, fax (978) 750-4470, or on the web at www.copyright.com. Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008, or online at <http://www.wiley.com/go/permission>.

Limit of Liability/Disclaimer of Warranty: While the publisher and author have used their best efforts in preparing this book, they make no representations or warranties with respect to the accuracy or completeness of the contents of this book and specifically disclaim any implied warranties of merchantability or fitness for a particular purpose. No warranty may be created or extended by sales representatives or written sales materials. The advice and strategies contained herein may not be suitable for your situation. You should consult with a professional where appropriate. Neither the publisher nor author shall be liable for any loss of profit or any other commercial damages, including but not limited to special, incidental, consequential, or other damages.

For general information on our other products and services or for technical support, please contact our Customer Care Department within the United States at (800) 762-2974, outside the United States at (317) 572-3993 or fax (317) 572-4002.

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic formats. For more information about Wiley products, visit our web site at www.wiley.com.

Library of Congress Cataloging-in-Publication Data is available.

ISBN: 978-1-119-05237-1

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

*This book is dedicated to my wife, Yan Wang,
and to my daughter, Susan (Xue) Bai.*

Contents

Preface	xxix
Acknowledgments	xxxii
Trademarks and Copyrights	xxxiii
Copyright Permissions	xxxv
About the Companion Website	xxxix
Chapter 1 Introduction to Microcontrollers and This Book	1
1.1 Microcontroller Configuration and Structure	2
1.2 The ARM® Cortex®M4 Microcontroller System	3
1.3 The TM4C123GH6PM Microcontroller Development Tools and Kits	4
1.4 Outstanding Features About This Book	5
1.5 Who This Book Is For	5
1.6 What This Book Covers	6
1.7 How This Book Is Organized and How to Use This Book	8
1.8 How to Use the Source Code and Sample Projects	9
1.9 Instructors and Customers Supports	11
Chapter 2 ARM® Microcontroller Architectures	13
2.1 Overview and Introduction	13
2.2 Introduction to ARM® Cortex®-M4 MCU	15
2.2.1 The Architecture of ARM® Cortex®-M4 MCU	17
2.2.1.1 The ARM® MCU Architecture	17
2.2.1.2 The Architecture of the ARM® Cortex®-M4 Core (CPU)	20
2.2.1.2.1 The Register Bank in the Cortex®-M4 Core	21
2.2.1.2.2 The Special Registers in the Cortex®-M4 Core	22
2.2.1.3 The Architecture of the Floating-Point Registers	25
2.3 The Memory Architecture	27
2.3.1 The Memory Map	28
2.3.2 The Stack Memory	29
2.3.3 The Program Models and States	32
2.3.4 The Memory Protection Unit (MPU)	33
2.4 The Nested Vectored Interrupt Controller (NVIC) Architecture	34
2.4.1 The Nested Vectored Interrupt Controller (NVIC) Features	35
2.4.2 Exception and Interrupt Sources	35
2.4.3 Exception Priority Levels and Mask Registers	35

viii Contents

2.4.4	Respond and Process Exceptions and Interrupts	36
2.4.5	Exception and Interrupt Vector Table	37
2.5	The Debug Architecture	37
2.6	Introduction to Tiva™ C Series ARM® Cortex®-M4 MCU-TM4C123GH6PM	38
2.6.1	TM4C123GH6PM Microcontroller Overview	39
2.6.2	TM4C123GH6PM Microcontroller On-Chip Memory Map	40
2.6.2.1	The System Peripherals	42
2.6.2.2	The On-Chip Peripherals	42
2.6.2.3	Interfaces to External Parallel Peripherals	44
2.6.2.4	Interfaces to External Serial Peripherals	44
2.6.3	TM4C123GH6PM Microcontroller General-Purpose Input–Output (GPIO) Module	44
2.6.3.1	The System Clock	45
2.6.3.2	The General Configuration Procedures for GPIO Peripherals	47
2.6.3.3	Tiva™ TM4C123GH6PM GPIO Architecture	47
2.6.3.3.1	The Port Control Register (GPIOPCTL)	49
2.6.3.3.2	The Data Control Registers	49
2.6.3.3.3	The Mode Control Registers	49
2.6.3.3.4	The Commit Control Registers	51
2.6.3.3.5	The Interrupt Control Registers	51
2.6.3.3.6	The Pad Control Registers	52
2.6.3.3.7	The Identification Registers	55
2.6.3.4	The Initialization and Configuration of TM4C123GH6PM GPIO Ports	55
2.6.4	TM4C123GH6PM Microcontroller System Controls	57
2.6.4.1	Device Identification	58
2.6.4.2	Reset Control	59
2.6.4.2.1	The Power-On Reset	60
2.6.4.2.2	The External Reset	61
2.6.4.2.3	The Brown-Out Reset (BOR)	61
2.6.4.2.4	The Software Reset	61
2.6.4.2.5	The Watchdog Timer Reset	62
2.6.4.3	Non-Maskable Interrupt Control	63
2.6.4.4	Clock Control	64
2.6.4.5	Other System Controls	67
2.6.4.5.1	The Run Mode	67
2.6.4.5.2	The Sleep Mode	68
2.6.4.5.3	The Deep-Sleep Mode	68
2.6.4.5.4	The Hibernate Mode	68
2.6.4.5.5	The System Timer (SysTick)	69
2.6.4.5.6	System Control Block (SCB)	70
2.6.4.6	System Clock Initialization and Configuration	71
2.7	Introduction to Tiva™ C Series LaunchPad™ TM4C123GXL Evaluation Board	72
2.8	Introduction to EduBASE ARM® Trainer	77
2.9	Chapter Summary	77
	Homework	79

Chapter 3 ARM® Microcontroller Development Kits**83**

3.1	Overview and Introduction	83
3.2	The Entire Tiva™ TM4C123G-based Development System	84
3.3	Download and Install Development Suite and Specified Firmware	86
3.4	Introduction to the Integrated Development Environment—Keil® MDK μVersion5	87
3.4.1	The Keil® MDK-ARM® for the MDK-Cortex-M Family	88
3.4.2	General Development Flow with MDK-ARM®	89
3.4.3	Warming Up Keil® MDK Cortex-M Kit with Example Projects	91
3.4.4	The Functions of the Keil® MDK-ARM® μVersion® 5 GUI	95
3.4.4.1	The File Menu	97
3.4.4.2	The Edit Menu	98
3.4.4.3	The Project Menu	101
3.4.4.4	The Flash Menu	121
3.4.4.5	The Debug Menu	121
3.4.4.6	The Peripherals Menu	123
3.4.4.7	The Tools Menu	124
3.4.4.8	The SVCS Menu	125
3.4.4.9	The Window Menu	126
3.4.4.10	The Help Menu	126
3.5	Embedded Software Development Procedure	127
3.6	The Keil® ARM®-MDK μVision5 Debugger and Debug Process	128
3.6.1	The ARM® μVision5 Debug Architecture	129
3.6.2	The ARM® Debug Adaptor and Debug Adaptor Driver	130
3.6.3	Tiva™ C Series LaunchPad™ Debug Adaptor and Debug Adaptor Driver	132
3.6.4	The ARM® μVersion5 Debug Process	133
3.6.5	The ARM® Trace Feature	134
3.6.5.1	Some Useful Trace Features Provided by Cortex®-M4 MCU	135
3.6.6	The ARM® Instruction Set Simulator	136
3.6.7	The ARM® Programs Running from SRAM	137
3.6.8	ARM® Optimizations	139
3.7	The TivaWare™ for C Series Software Suite	140
3.7.1	The TivaWare™ C Series Software Package	142
3.7.1.1	The Peripheral Driver Library (DriverLib)	143
3.7.1.2	The Boot Loader	144
3.7.1.3	The Utilities	144
3.7.2	TivaWare™ C Series for TM4C123G LaunchPad™ Evaluation Kit	145
3.7.2.1	TivaWare™ C Series LaunchPad™ Evaluation Software Package	145
3.8	The TivaWare™ for C Series Utilities and Other Supports	147
3.8.1	Additional Utilities Provided by TivaWare™ for C Series	148
3.8.1.1	The LMFlash Programmer	148
3.8.1.2	The UniFlash	149
3.8.1.3	The FTDI Drivers	149
3.8.1.4	The IQMath Library	149
3.8.1.5	TivaWare™ for C Series CMSIS Support	150

3.9	Program Examples	151
3.10	Chapter Summary	152
	Homework	152

Chapter 4 ARM® Microcontroller Software and Instruction Set 155

4.1	Overview and Introduction	155
4.2	Introduction to ARM® Cortex®-M4 Software Development Structure	156
4.3	Introduction to ARM® Cortex®-M4 Assembly Instruction Set	157
4.3.1	The ARM® Cortex®-M4 Assembly Language Syntax	158
4.3.2	The ARM® Cortex®-M4 Pseudo Instructions	160
4.3.3	The ARM® Cortex®-M4 Addressing Modes	161
4.3.3.1	The Immediate Offset Addressing Mode	162
4.3.3.1.1	Regular Immediate Offset Addressing Mode	162
4.3.3.1.2	Pre-Indexed Immediate Offset Addressing Mode	163
4.3.3.1.3	Post-Indexed Immediate Offset Addressing Mode	163
4.3.3.1.4	Regular Immediate Offset Addressing Mode with Unprivileged Access	163
4.3.3.2	The Register Offset Addressing Mode	164
4.3.3.3	The PC-Relative Addressing Mode	165
4.3.3.4	Load and Store Multiple Registers Addressing Mode	167
4.3.3.5	PUSH and POP Register Addressing Mode	170
4.3.3.6	Load and Store Register Exclusive Addressing Mode	170
4.3.3.7	Inherent Addressing Mode	171
4.3.3.8	Addressing Mode Summary	171
4.3.4	The ARM® Cortex®-M4 Instruction Set Categories	172
4.3.4.1	Data Moving Instructions	172
4.3.4.2	Arithmetic Instructions	174
4.3.4.3	Logic Instructions	176
4.3.4.4	Shift and Rotate Instructions	178
4.3.4.5	Data Conversion Instructions	179
4.3.4.6	Bit-Field Processing Instructions	182
4.3.4.7	Compare and Test Instructions	186
4.3.4.8	Program Flow Control Instructions	187
4.3.4.9	Saturation Instructions	191
4.3.4.10	Exception-Related Instructions	193
4.3.4.11	Sleep Mode Instructions	194
4.3.4.12	Memory Barrier Instructions	194
4.3.4.13	Miscellaneous Instructions	195
4.3.4.14	Unsupported Instructions	196
4.4	ARM® Cortex®-M4 Software Development Procedures	196
4.5	Using C Language to Develop ARM® Cortex®-M4 Microcontroller Applications	197
4.5.1	The Standard Data Types Used in Intrinsic Functions	198
4.5.2	The CMSIS-Core-Specific Intrinsic Functions	200
4.5.3	The Keil® ARM® Compiler-Specific Intrinsic Functions	202
4.5.4	Inline Assembler	204

4.5.5	Idiom Recognition	205
4.5.6	C Programming Development Guideline and Procedure	206
4.5.6.1	Organization of the C Program Files	207
4.5.6.2	The Header Files	208
4.5.6.3	The Implementation Files	209
4.5.6.4	The Application Files	211
4.5.6.5	Naming Convention and Definition	211
4.5.7	The TivaWare™ Peripheral Driver Library	213
4.5.7.1	The Programming Models	213
4.5.7.2	The Direct Register Access Model	214
4.5.7.2.1	The Hardware Architecture of the Example Project	214
4.5.7.2.2	The Structure and Bit Function of System Control and GPIO Registers	215
4.5.7.2.3	The Symbolic Definitions and Macros	218
4.5.7.2.4	The Programming Operations for Symbolic Definitions	219
4.5.7.2.5	Develop a Sample Project Using the DRA Model	220
4.5.7.3	The Peripheral Driver Library and API Functions	224
4.5.7.3.1	System Control API Functions	225
4.5.7.3.2	GPIO API Functions	229
4.5.7.3.3	Develop a Sample Project Using the SD Model	232
4.5.7.4	A Comparison Between Two Programming Models	238
4.5.7.5	A Combined Programming Model Example	238
4.5.7.5.1	Create the Header File	239
4.5.7.5.2	Create the C Source File	240
4.5.7.5.3	Include System Header Files and Add Static Library into the Project	241
4.5.7.5.4	Compile and Link Project to Create the Image or Executable File	242
4.6	Chapter Summary	243
	Homework	244

Chapter 5 ARM® Microcontroller Interrupts and Exceptions **261**

5.1	Overview and Introduction	261
5.2	Exceptions and Interrupts in the ARM® Cortex®-M4 MCU System	263
5.2.1	Exception and Interrupt Types	265
5.2.2	Exceptions and Interrupts Management	265
5.2.3	Exception and Interrupt Processing	268
5.2.3.1	Exception and Interrupt Inputs and Pending Status	269
5.2.3.2	Exception and Interrupt Vector Table	270
5.2.3.3	Definitions of the Priority Levels	271
5.3	Exceptions and Interrupts in the TM4C123GH6PM Microcontroller System	273
5.3.1	Local Interrupt Configurations and Controls for GPIO Pins	273
5.3.1.1	Initialize and Configure GPIO Interrupt Control Registers	274
5.3.2	Local Interrupt Configurations and Controls for GPIO Ports	276
5.3.2.1	The NVIC Interrupt Priority-Level Registers	276
5.3.2.2	The NVIC Interrupt Set Enable Registers	280

xii Contents

5.3.3	Global Interrupt Configurations and Controls	281
5.3.4	The Vector Table and Vectors Used in the TM4C123GH6PM MCU	282
5.3.5	The GPIO Interrupt Handling and Processing Procedure	284
5.4	Developing GPIO Port Interrupt Projects to Handle GPIO Interrupts	285
5.4.1	Two Software Packages Used in the TM4C123GH6PM MCU System	286
5.4.1.1	The TivaWare™ Software Package (TWSP)	286
5.4.1.1.1	Two Header Files Used in the TM4C123GH6PM MCU System	288
5.4.1.1.2	The Register Driver Definition Header File in the TivaWare™ Software Package	289
5.4.1.1.3	The CMSIS Cortex-M4 Peripheral Layer Header File for TM4C123GH6PM	289
5.4.1.2	The CMSIS Core Software Package (CMSISCSP)	290
5.4.2	Using DRA Programming Model to Handle GPIO Interrupts	290
5.4.2.1	Create a New Project GPIOInt and the Header File	291
5.4.2.2	Create a New C Code File GPIOInt and Add It into the Project	292
5.4.2.3	Set Up the Environment to Compile and Link the Project	294
5.4.3	Using CMSIS Core Macros for NVIC Registers to Handle GPIO Interrupts	294
5.4.3.1	Popular Data Structures Defined in the CMSIS Core Header File	295
5.4.3.2	IRQ Numbers Defined in the TivaWare™ System Header File	297
5.4.3.3	The NVIC Macros Defined in the TivaWare™ System Header Files	299
5.4.3.4	The NVIC Structure Defined in the CMSIS Core Header File	300
5.4.3.5	Building Sample Project to Use CMSIS Core Macros for NVIC to Handle Interrupts	302
5.4.3.6	Create a New Project NVICInt and Add the C Code File	303
5.4.4	Using TivaWare™ Peripheral Driver Library API Functions to Handle GPIO Interrupts	306
5.4.4.1	NVIC API Functions Defined in the TivaWare™ Peripheral Driver Library	306
5.4.4.2	GPIO Interrupt-Related API Functions in the TivaWare™ Peripheral Driver Library	308
5.4.4.3	Building Sample Project to Use Peripheral Driver Library to Handle Interrupts	309
5.4.4.4	Create a New Project SDInt and Add the C Code File	310
5.4.4.5	Configuring the Environments and Run the Project	312
5.4.5	Using CMSIS Core Access Functions to Handle GPIO Interrupts	313
5.4.5.1	Building Sample Project to Use CMSIS Core Functions to Handle Interrupts	314
5.4.5.2	Create a New Project CMSISInt and Add the C Code File	314
5.4.5.3	Configure the Environments and Run the Project	316
5.5	Comparison Among Four Interrupt Programming Methods	317
5.6	Chapter Summary	318
	Homework	319

Chapter 6 ARM® Microcontroller Memory System 333

6.1	Overview and Introduction	333
6.2	Memory Architecture in the TM4C123GH6PM MCU System	334
6.2.1	Static Random Access Memory (SRAM)	336
6.2.2	Flash Memory	336
6.2.2.1	Basic Operations of the Flash Memory	337
6.2.2.2	The 32-Word Flash Memory Write Buffer	338
6.2.2.3	Flash Control Registers	339
6.2.2.4	Boot Configuration Register (BOOTCFG)	339
6.2.2.5	Flash Memory Address Register (FMA)	341
6.2.2.6	Flash Memory Data Register (FMD)	342
6.2.2.7	Flash Memory Control Register (FMC)	342
6.2.2.8	Flash Memory Control 2 Register (FMC2)	342
6.2.2.9	The Flash Write Buffer Valid Register (FWBVAL)	343
6.2.2.10	Flash Controller Raw Interrupt Status Register (FCRIS)	344
6.2.2.11	Flash Controller Interrupt Mask Register (FCIM)	346
6.2.2.12	Flash Controller Masked Interrupt Status and Clear Register (FCMISC)	346
6.2.2.13	Other Control Registers Related to Flash Memory Control	349
6.2.3	Flash Memory Protection Control	349
6.2.4	Internal Read-Only Memory (ROM)	351
6.2.4.1	The Boot Loader	352
6.2.4.2	The TivaWare™ Peripheral Driver Library	352
6.2.4.3	The ROM Control Register (RMCTL)	354
6.2.4.4	The ROM Software Map Register (ROMSWMAP)	354
6.2.5	Electrical Erased Programmable Read-Only Memory (EEPROM)	354
6.2.5.1	EEPROM Initialization and Configuration	356
6.2.5.2	Most Important Control Registers Used in the EEPROM Module	357
6.2.5.2.1	The EEPROM Current Block Register (EEBLOCK)	357
6.2.5.2.2	The EEPROM Current Offset Register (EEOFFSET)	357
6.2.5.2.3	EEPROM Done Status Register (EEDONE)	357
6.2.5.2.4	EEPROM Support Control and Status Register (EESUPP)	359
6.2.5.2.5	EEPROM Protection Register (EEPROT)	359
6.2.5.3	Other Important Control Registers Used in the EEPROM Module	360
6.3	Memory Map in TM4C123GH6PM MCU System	361
6.4	Bit-Band Operations	362
6.4.1	The Mapping Relationship Between the Bit-Band Region and the Bit-Band Alias Region	365
6.4.2	The Advantages of Using the Bit-Band Operations	365
6.4.3	An Illustration Example of Using Bit-Band Alias Addresses	367
6.4.4	Bit-Band Operations for Different Data Sizes	369
6.4.5	Bit-Band Operations Built in C Programs	369
6.5	Memory Requirements and Memory Properties	370
6.5.1	Memory Requirements	371
6.5.2	Memory Access Attributes	372

6.5.3	Memory Endianness	373
6.5.3.1	The Little Endian Format	374
6.5.3.2	The Big Endian Format	374
6.6	Memory System Programming Methods	375
6.6.1	The API Functions Used for Flash Memory Programming	376
6.6.2	The API Functions Used for EEPROM Programming	378
6.7	Memory System Programming Projects	380
6.7.1	Flash Memory Programming	380
6.7.1.1	Programming Flash Memory for Multiple Words with DRA Method (Polled)	380
6.7.1.1.1	The Operational Sequence of the Programming Flash Memory	380
6.7.1.1.2	The Programming Macros for Flash Memory Registers and Parameters	381
6.7.1.1.3	Build the Project to Program Multiple Words for Flash Memory	382
6.7.1.1.4	Build and Run the Project to Perform Erase and Write Operations	386
6.7.1.2	Programming Flash Memory for Multiple Words with the DRA Method (Interrupt Driven)	388
6.7.1.2.1	The Erase and Write Interrupts Processing Procedure	389
6.7.1.2.2	Special Features Utilized in the Project	390
6.7.1.2.3	Build the Project to Program Multiple Words for Flash Memory with Interrupts	390
6.7.1.2.4	Set Up the Environment to Build and Run the Project	396
6.7.1.3	Programming Flash Memory for Buffered Words with the DRA Method	397
6.7.1.3.1	The Buffer Words Programming Procedure	397
6.7.1.3.2	Develop the Buffer Words Programming Project DRAFlashBuffer	398
6.7.1.3.3	Build and Set Up the Environment to Run the Project	400
6.7.2	EEPROM Programming	401
6.7.2.1	Special Features in the EEPROM Programming Process	401
6.7.2.2	EEPROM Programming Operational Sequence	402
6.7.2.2.1	Configure and Set Up EEBLOCK and EEOFFSET Registers	403
6.7.2.2.2	Implement and Update the EEBLOCK and EEOFFSET Registers	404
6.7.3	Three Kinds of System Header Files in the TM4C123GH6PM MCU System	405
6.7.3.1	The Register Driver Definitions Header File TM4C123GH6PM.h	405
6.7.3.2	The CMSIS Cortex [®] -M4 Peripheral Hardware Layer Header File TM4C123GH6PM.h	406
6.7.3.3	System Header Files for All Internal Peripherals and System Control Devices	406
6.7.3.4	Enable the EEPROM Module in Run Mode and Reset EEPROM	407

6.7.4	Build Example EEPROM Programming Projects	408
6.7.4.1	Programming EEPROM with the DRA Method (Polling-Driven)	408
6.7.4.1.1	Create the Header File DRAEEPROMPoll.h	408
6.7.4.1.2	Create the Source File DRAEEPROMPoll.c	409
6.7.4.1.3	Set Up the Environment to Build and Run the Project	413
6.7.4.2	Programming EEPROM with the DRA Method (Interrupt-Driven)	414
6.7.4.2.1	Modify the Header File DRAEEPROMInt.h	416
6.7.4.2.2	Modify the Source File DRAEEPROMInt.c	417
6.7.4.2.3	Set Up the Environment to Build and Run the Project	419
6.8	Chapter Summary	420
	Homework	421

Chapter 7 ARM® Cortex®-M4 Parallel I/O Ports Programming **433**

7.1	Overview and Introduction	433
7.2	GPIO Module Architecture and GPIO Port Configuration	434
7.3	GPIO Port Control Registers	437
7.3.1	GPIO Port Initialization and Configuration	438
7.4	On-Board Keypad Interface Programming Project	440
7.4.1	The Keypad Interfacing Programming Structure	441
7.4.2	Create the Keypad Interfacing Programming Project (Polling-Driven)	442
7.4.2.1	Create the C Source File DRAKeyPadPoll.c	443
7.4.3	Set Up the Environment to Build and Run the Project	446
7.5	Analog-to-Digital Converter Programming Project	446
7.5.1	ADC Modules in the TM4C123GH6PM MCU System	446
7.5.2	ADC Module Architecture and Functional Block Diagram	447
7.5.3	ADC Module Components and Signal Descriptions	448
7.5.3.1	Analog Input Signals and GPIO Analog Control Registers	449
7.5.3.1.1	GPIO Alternate Function Select (GPIOAFSEL) Register	450
7.5.3.1.2	GPIO Digital Enable (GPIODEN) Register	450
7.5.3.1.3	GPIO Analog Mode Select (GPIOAMSEL) Register	451
7.5.3.2	Sample Sequencer Controls and Their Control Registers	451
7.5.3.2.1	ADC Sample Sequencer Input Multiplexer Select (ADCSSMUXn) Register	452
7.5.3.2.2	ADC Sample Sequencer Control (ADCSSCTLn) Register	454
7.5.3.2.3	ADC Active Sample Sequencer (ADCACTSS) Register	458
7.5.3.2.4	ADC Processor Sample Sequencer Initiate (ADCPSSI) Register	459
7.5.3.2.5	ADC Sample Sequencer Result FIFO (ADCSSFIFOn) Register	460
7.5.3.3	ADC Module Control Functions and Related Registers	461
7.5.3.3.1	ADC Module Clocking	461
7.5.3.3.2	ADC Interrupt Request and Handling	463
7.5.3.3.3	Sampling Events and Trigger Sources	467
7.5.3.3.4	DMA Operations	470
7.5.4	Analog-to-Digital Converter	470

7.5.4.1	Voltage Reference and Resolutions	471
7.5.4.2	Differential Input Mode	471
7.5.4.3	Internal Temperature Sensor	472
7.5.5	Initialization and Configuration	473
7.5.5.1	ADC-Related GPIO Ports Initialization	473
7.5.5.2	ADC Module Initialization	474
7.5.5.3	Sample Sequencers Initialization	474
7.5.6	Build the Analog-to-Digital Converter Programming Project	475
7.5.6.1	ADC Module in EduBASE ARM® Trainer	475
7.5.6.2	Create the ADC Programming Project (Polling-Driven)	476
7.5.6.3	Create the Source File DRAADCPoll.c	476
7.5.6.4	Set Up the Environment to Build and Run the Project	479
7.5.7	ADC Module API Functions Provided in the TivaWare™ Peripheral Driver Library	480
7.5.7.1	Configuring and Handling the Sample Sequencers API Functions	481
7.5.7.2	Configuring and Controlling the Processor Trigger API Functions	481
7.5.7.3	Configuring and Processing the ADC Interrupt API Functions	483
7.5.7.4	Build an Example ADC Project Using API Functions	484
7.6	PWM-Controlled DC and Step Motors Programming Project	486
7.6.1	The PWM Principle and Implementations	487
7.6.2	PWM Modules in the TM4C123GH6PM MCU System	487
7.6.2.1	The PWM Generator Block	488
7.6.2.1.1	The PWM Counter (Timer)	488
7.6.2.1.2	The PWM Comparators	488
7.6.2.1.3	The PWM Output Signals Generator	489
7.6.2.1.4	The Dead-Band Generator	490
7.6.3	PWM Generator Functional Block Diagram	490
7.6.3.1	PWM Generator Block Control Register (PWMnCTL)	491
7.6.3.2	PWM Generator Block Load Register (PWMnLOAD)	493
7.6.3.3	PWM Generator Block Count Register (PWMnCOUNT)	493
7.6.3.4	PWM Generator Block Comparator A Register (PWMnCMPA)	493
7.6.3.5	PWM Generator Block Comparator B Register (PWMnCMPB)	494
7.6.3.6	PWM Generator A Register (PWMnGENA)	494
7.6.3.7	PWM Generator B Register (PWMnGENB)	495
7.6.3.8	PWM Generator Dead-Band Control Register (PWMnDBCTL)	496
7.6.3.9	PWM Generator Dead-Band Rising-Edge Delay Register (PWMnDBRISE)	497
7.6.3.10	PWM Generator Dead-Band Falling-Edge Delay Register (PWMnDBFALL)	497
7.6.3.11	PWM Interrupt and Trigger Enable Register (PWMnINTEN)	498
7.6.3.12	PWM Raw Interrupt Status Register (PWMnRIS)	498
7.6.3.13	PWM Interrupt Status and Clear Register (PWMnISC)	499
7.6.3.14	PWM Fault Source n Register (PWMFLTSRCn)	501
7.6.4	PWM Module Architecture and Functional Block Diagram	502
7.6.4.1	The Control and Status Block	503
7.6.4.1.1	The Run-Mode Clock Configuration Register (RCC)	504
7.6.4.1.2	The PWM Master Control Register (PWMCTL)	504

7.6.4.1.3	The PWM Timer Base Synchronous Register (PWMSYNC)	504
7.6.4.1.4	The PWM Status Register (PWMSTATUS)	505
7.6.4.1.5	The PWM Peripheral Properties Register (PWMPP)	505
7.6.4.2	The Output Control Block	505
7.6.4.2.1	The PWM Output Enable Register (PWMENABLE)	505
7.6.4.2.2	The PWM Output Inversion Register (PWMINVERT)	506
7.6.4.2.3	The PWM Output Fault Register (PWMFAULT)	506
7.6.4.2.4	The PWM Fault Condition Value Register (PWMFAULTVAL)	507
7.6.4.2.5	The PWM Enable Update Register (PWMENUUPD)	507
7.6.4.3	The Interrupt Control Block	507
7.6.4.3.1	The PWM Interrupt Enable Register (PWMINTEN)	508
7.6.4.3.2	The PWM Raw Interrupt Status Register (PWMRIS)	509
7.6.4.3.3	The PWM Interrupt Status and Clear Register (PWMISC)	509
7.6.5	PWM Module Components and Signal Descriptions	509
7.6.5.1	PWM Signal Description	510
7.6.5.2	Synchronization Methods	511
7.6.5.3	Fault Conditions	512
7.6.6	PWM Module Initialization and Configuration	513
7.6.6.1	Initialize and Configure the Clock Source for PWM Module and GPIO Ports	513
7.6.6.2	Initialize and Configure GPIO Ports and Pins Related to PWM Modules	513
7.6.6.3	Initialize and Configure the PWM Module and Generators	514
7.6.7	PWM Module Architecture in the EduBASE ARM® Trainer	515
7.6.8	Build an Example PWM Programming Project	516
7.6.8.1	Create a PWM Application Project DRAPWM	517
7.6.8.2	Set Up the Environment to Build and Run the Project	520
7.7	The PWM API Functions in the TivaWare™ Peripheral Driver Library	521
7.7.1	PWM Modules and Generators Configuration and Set Up Control Functions	521
7.7.2	PWM Output Control Functions	523
7.7.3	PWM Interrupt and Fault Control Functions	523
7.8	Chapter Summary	525
	Homework	527

Chapter 8 ARM® Cortex®-M4 Serial I/O Ports Programming **547**

8.1	Overview and Introduction	547
8.2	GPIO Module Architecture and GPIO Port Configuration	548
8.3	Synchronous Serial Interface (SSI)	551
8.3.1	Asynchronous and Synchronous Communication Protocols and Data Framing	552
8.3.2	Synchronous Serial Interface Architecture and Functional Block Diagram	555
8.3.3	The Synchronous Data Transmission Format and Frame	556
8.3.3.1	Texas Instruments™ Synchronous Serial Frame	558
8.3.3.2	Freescale SPI Frame	558

8.3.3.3	MICROWIRE Frame	559	
8.3.4	SSI Module Components and Signal Descriptions	560	
8.3.4.1	SSI Control Signals and GPIO SSI Control Registers	560	
8.3.4.2	SSI Module Bit Rate Generation and Clock Control	562	
8.3.4.3	SSI Module Control/Status and FIFO Control	564	
8.3.4.3.1	SSI Control 1 Register (SSICR1)	564	
8.3.4.3.2	SSI Status Register (SSISR)	565	
8.3.4.3.3	SSI Data Register (SSIDR)	565	
8.3.4.3.4	FIFO Operations	566	
8.3.4.4	SSI Module Interrupt and DMA Control	567	
8.3.4.4.1	SSI Interrupt Mask Register (SSIIM)	568	
8.3.4.4.2	SSI Raw Interrupt Status Register (SSIRIS)	569	
8.3.4.4.3	SSI DMA Control Register (SSIDMACTL)	569	
8.3.4.5	SSI Module Transmit/Receive Logic Control	570	
8.3.4.6	SSI Modules Initialization and Configurations	570	
8.3.4.6.1	SSI-Module-Related GPIO Ports Initialization	570	
8.3.4.6.2	SSI Module Initialization and Configuration	571	
8.3.4.6.3	SSI Module Clock Source and Bit Rate Initialization and Configuration	571	
8.3.5	Build the On-Board LCD Interface Programming Project	572	
8.3.5.1	SSI Module Interface for the LCD in EduBASE ARM® Trainer	572	
8.3.5.2	The Serial Shift Register 74VHCT595	573	
8.3.5.3	The LCD Module TC1602A and LCD Controller SPLC780	574	
8.3.5.3.1	Interfacing Control Signals Between the MCU and the SPLC780	576	
8.3.5.3.2	Control and Interface Programming for SPLC780	578	
8.3.5.3.3	LCD Programming Instruction Structure and Sequence	580	
8.3.5.4	Build the Example LCD Interfacing Project	583	
8.3.5.4.1	Create a Direct Register Access LCD Project DRALCD	584	
8.3.5.4.2	Create the Header File DRALCD.h	584	
8.3.5.4.3	Create the C Source File DRALCD.c	585	
8.3.5.4.4	Set Up the Environment to Build and Run the Project	589	
8.3.6	Build On-Board 7-Segment LED Interface Programming Project	589	
8.3.6.1	Structure of 7-Segment LEDs	589	
8.3.6.2	SSI Module Interface for the 7-Segment LED in the EduBASE ARM® Trainer	590	
8.3.6.3	Build the Example LED Interfacing Project	592	
8.3.6.3.1	Create a Direct Register Access LED Project DRALED	593	
8.3.6.3.2	Create the C Source File DRALED.c	593	
8.3.6.3.3	Set Up the Environment to Build and Run the Project	595	
8.3.7	Build Digital-to-Analog Converter Programming Project	595	
8.3.7.1	SSI Module Interface for the DAC-MCP4922 in the EduBASE ARM® Trainer	595	
8.3.7.2	The Operations and Programming for MCP4922 DAC	596	
8.3.7.3	The Analog-to-Digital Converter TLC-548	598	
8.3.7.4	Build the Example DAC Interfacing Project	599	
8.3.7.4.1	Create a Direct Register Access DAC Project DRADAC	600	

8.3.7.4.2	Create the Header File DRADAC.h	600
8.3.7.4.3	Create the C Source File DRADAC.c	600
8.3.7.4.4	Set Up the Environment to Build and Run the Project	603
8.3.8	SSI API Functions Provided by TivaWare™ Peripheral Driver Library	604
8.3.8.1	The SSI Module Initialization and Configuration Functions	604
8.3.8.2	The SSI Module Control and Status Functions	605
8.3.8.3	The SSI Module Data Processing Functions	606
8.3.8.4	The SSI Module Interrupt Source and Processing Functions	607
8.3.8.5	Build an Example Project to Interface Serial Peripherals Using the SSI Module	608
8.3.8.5.1	Create a New Software Driver Model Project SDLCD	608
8.3.8.5.2	Create the Header File SDLCD.h	608
8.3.8.5.3	Create the C Source File SDLCD.c	608
8.4	Inter-Integrated Circuit (I2C) Interface	611
8.4.1	I2C Module Bus Configuration and Operational Status	612
8.4.2	I2C Module Architecture and Functional Block Diagram	613
8.4.3	I2C Module Data Transfer Format and Frame	614
8.4.4	I2C Module Operational Sequence	614
8.4.4.1	I2C Module Works in the Master Transmit Mode	614
8.4.4.2	I2C Module Works in the Master Receive Mode	616
8.4.4.3	I2C Module Works in the Slave Transmit and Receive Modes	616
8.4.5	I2C Module Major Operational Components and Control Signals	618
8.4.6	I2C Module Running Speeds (Clock Rates) and Interrupts	620
8.4.6.1	I2C Module High-Speed Mode	621
8.4.6.2	I2C Module Interrupts Generation and Processing	621
8.4.6.2.1	I2C Master Interrupts	622
8.4.6.2.2	I2C Slave Interrupts	622
8.4.7	I2C Interface Control Signals and GPIO I2C Control Registers	622
8.4.8	I2C Module Control Registers and Their Functions	623
8.4.8.1	I2C Module Master Control Registers	623
8.4.8.1.1	I2C Master Slave Address Register (I2CMSA)	623
8.4.8.1.2	I2C Master Control/Status Register (I2CMCS)	624
8.4.8.1.3	I2C Master Data Register (I2CMDR)	624
8.4.8.1.4	I2C Master Timer Period Register (I2CMTPR)	624
8.4.8.1.5	I2C Master Configuration Register (I2CMCR)	625
8.4.8.1.6	I2C Master Clock Low Timeout Count Register (I2CMCLKOCNT)	625
8.4.8.1.7	I2C Master Bus Monitor Register (I2CMBMON)	626
8.4.8.1.8	I2C Master Interrupt Mask Register (I2CMIMR)	626
8.4.8.1.9	I2C Master Raw Interrupt Status Register (I2CMRIS)	626
8.4.8.1.10	I2C Master Masked Interrupt Status Register (I2CMMIS)	626
8.4.8.1.11	I2C Master Interrupt Clear Register (I2CMICR)	627
8.4.8.2	I2C Module Slave Control Registers	627
8.4.8.2.1	I2C Slave Own Address Register (I2CSOAR)	627
8.4.8.2.2	I2C Slave Control Status Register (I2CSCSR)	627
8.4.8.2.3	I2C Slave Data Register (I2CSDR)	628
8.4.8.2.4	I2C Slave Own Address 2 Register (I2CSOAR2)	628

8.4.8.2.5	I2C Slave ACK Control Register (I2CSACKCTL)	628
8.4.8.2.6	I2C Slave Interrupt Mask Register (I2CSIMR)	629
8.4.8.2.7	I2C Slave Raw Interrupt Status Register (I2CSRIS)	629
8.4.8.2.8	I2C Slave Masked Interrupt Status Register (I2CSMIS)	629
8.4.8.2.9	I2C Slave Interrupt Clear Register (I2CSICR)	629
8.4.9	I2C Module Initializations and Configurations	630
8.4.9.1	Initializations and Configurations for the I2C-Related GPIO Pins	630
8.4.9.2	Initializations and Configurations for the I2C Module	630
8.4.10	Build an Example I2C Module Project	631
8.4.10.1	The BQ32000 Real Time Clock (RTC)	631
8.4.10.2	The Interface Between the BQ32000 and EduBASE ARM® Trainer	633
8.4.10.3	Create a DRA Model I2C Project DRAI2C	634
8.4.10.4	Create the Source File DRAI2C	634
8.4.10.5	Set Up the Environment to Build and Run the Project	638
8.4.11	I2C API Functions Provided by TivaWare™ Peripheral Driver Library	639
8.4.11.1	Master Operations	639
8.4.11.2	I2C Module Status and Initialization API Functions	640
8.4.11.3	I2C Module Sending and Receiving Data API Functions	641
8.5	Universal Asynchronous Receivers/Transmitters (UARTs)	642
8.5.1	Asynchronous Serial Communication Protocols and Data Framing	642
8.5.2	Asynchronous Serial Interface Architecture and Functional Block Diagram	643
8.5.3	UART Module Operations and Control Registers	645
8.5.3.1	Transmit/Receive Logic and Data Transmission and Receiving	645
8.5.3.2	UART Modem Handshake Support	645
8.5.3.3	UART FIFO Operations	647
8.5.3.4	UART Interrupts and DMA Control	648
8.5.3.5	UART Serial IR (SIR) Support	649
8.5.3.6	9-Bit UART Mode	649
8.5.3.7	UART Module Clock Control and Baud Rate Generation Registers	650
8.5.3.8	UART Module Control/Status and FIFO Control Registers	651
8.5.3.8.1	UART Control Register (UARTCTL)	651
8.5.3.8.2	UART Line Control Register (UARTLCRH)	653
8.5.3.8.3	UART Receive Status/Error Clear Register (UARTRSR/UARTECR)	653
8.5.3.8.4	UART Data Register (UARTDR)	654
8.5.3.8.5	UART Flag Register (UARTFR)	655
8.5.3.9	UART Module Interrupt and DMA Control Registers	655
8.5.3.9.1	UART Interrupt FIFO Level Select (UARTIFLS) Register	656
8.5.3.9.2	UART Raw Interrupt Status (UARTRIS) Register	656
8.5.3.9.3	UART Interrupt Mask (UARTIM) Register	656
8.5.3.9.4	UART Masked Interrupt Status (UARTMIS) Register	657
8.5.3.9.5	UART Interrupt Clear Register (UARTICR)	657
8.5.3.9.6	UART DMA Control (UARTDMACTL) Register	657
8.5.4	UART Module Control Signals and Related GPIO Pins	658

8.5.5	UART Module Initializations and Configurations	659
8.5.5.1	Initialize and Configure the UART-Related GPIO Ports and Pins	659
8.5.5.2	Initialize and Configure Clock Source and Baud Rate for the UART Module	659
8.5.5.3	Initialize and Configure the UART Module	660
8.5.6	Build an Example UART Module Project	660
8.5.6.1	Create a New UART Module Project DRAUART	661
8.5.6.2	Create a New C Source File	661
8.5.6.3	Set Up the Environment to Build and Run the Project	664
8.5.7	The UART API Functions Provided by the TivaWare™ Peripheral Driver Library	664
8.5.7.1	Clock Source for the Baud Rate Generator API Functions	665
8.5.7.2	Configure and Control the UART Modules API Functions	666
8.5.7.3	UART Send and Receive Data API Functions	667
8.5.7.4	UART Interrupt Handling API Functions	667
8.6	Chapter Summary	668
	Homework	669

Chapter 9 ARM® Cortex®-M4 Timer and USB Programming **691**

9.1	Overview and Introduction	691
9.2	General-Purpose Timers	692
9.2.1	The GPTM Architecture and Functional Block Diagram	693
9.2.2	The General-Purpose Timer Module Components	694
9.2.2.1	Prescaler Registers	695
9.2.2.2	Match Registers	695
9.2.2.3	Shadow Registers	695
9.2.3	The General-Purpose Timer Module Operational Modes	695
9.2.3.1	One-Shot and Periodic Timer Mode	696
9.2.3.2	Periodic Snapshot Timer Mode	698
9.2.3.3	Wait-for-Trigger Mode	699
9.2.3.4	Real-Time Clock Timer Mode	699
9.2.3.5	Input Edge-Count Mode	699
9.2.3.6	Input Edge-Time Mode	700
9.2.3.7	PWM Mode	702
9.2.3.8	DMA Mode	703
9.2.3.9	Synchronizing GP Timer Blocks	703
9.2.3.10	Concatenated Modes	703
9.2.4	The General-Purpose Timer Module Registers	704
9.2.4.1	Timer A Control Register Group	704
9.2.4.1.1	GPTM Configuration Register (GPTMCFG)	705
9.2.4.1.2	GPTM Control Register (GPTMCTL)	705
9.2.4.1.3	GPTM Timer A Mode Register (GPTMTAMR)	705
9.2.4.1.4	GPTM Timer A Interval Load Register (GPTMTAILR)	705
9.2.4.1.5	GPTM Timer A Match Register (GPTMTAMATCHR)	706
9.2.4.1.6	GPTM Timer A Prescale Register (GPTMTAPR)	707

9.2.4.1.7	GPTM Timer A Prescale Match Register (GPTMTAPMR)	708
9.2.4.1.8	GPTM Timer A Prescale Snapshot Register (GPTMTAPS)	708
9.2.4.2	Timer A Status Register Group	708
9.2.4.2.1	GPTM Timer A Register (GPTMTAR)	708
9.2.4.2.2	GPTM Timer A Value Register (GPTMTAV)	709
9.2.4.2.3	GPTM Timer A Prescale Value Register (GPTMTAPV)	709
9.2.4.3	Timers A and B Interrupt and Configuration Register Group	709
9.2.4.3.1	GPTM Interrupt Mask Register (GPTMIMR)	710
9.2.4.3.2	GPTM Raw Interrupt Status Register (GPTMRIS)	711
9.2.4.3.3	GPTM Masked Interrupt Status Register (GPTMMIS)	711
9.2.4.3.4	GPTM Interrupt Clear Register (GPTMICR)	711
9.2.4.3.5	GPTM Synchronize Register (GPTMSYNC)	711
9.2.4.3.6	GPTM Peripheral Properties Register (GPTMPP)	711
9.2.5	The General-Purpose Timer Module GPIO-Related Control Signals	712
9.2.6	The General-Purpose Timer Module Initializations and Configurations	713
9.2.6.1	Initialization and Configuration for One-Shot/Periodic Timer Mode	714
9.2.6.2	Initialization and Configuration for Input Edge-Count Mode	714
9.2.6.3	Initialization and Configuration for Input Edge-Time Mode	715
9.2.6.4	Initialization and Configuration for Real-Time Clock (RTC) Mode	716
9.2.6.5	Initialization and Configuration for PWM Mode	716
9.2.7	Build an Example General Purpose Timer Project	717
9.2.8	Popular Implementations on GPTM Modules	718
9.2.8.1	Input Edge-Count Implementations	719
9.2.8.2	Input Edge-Time Implementations	721
9.2.8.3	PWM Implementations	723
9.2.9	The API Functions Used for General-Purpose Timer Module	727
9.2.9.1	The API Functions Used for GPTM Module Configurations and Controls	727
9.2.9.2	The API Functions Used for GPTM Module Contents and Related Operations	727
9.2.9.3	The API Functions Used for GPTM Module Interrupt Handling	730
9.2.9.4	An Implementation of Using Timer API Functions to Measure PWM Pulses	731
9.3	Watchdog Timers	732
9.3.1	The Watchdog Timer Architecture and Functional Block Diagram	734
9.3.2	The Watchdog Timer Operational Sequence and Timing Access	735
9.3.3	The Watchdog Timer Registers	735
9.3.3.1	The Watchdog Module Control and Content Registers	735
9.3.3.1.1	Watchdog Timer Control Register (WDTCTL)	736
9.3.3.1.2	Watchdog Timer Load Register (WDTLOAD)	736
9.3.3.1.3	Watchdog Timer Value Register (WDTVALUE)	736
9.3.3.1.4	Watchdog Timer Lock Register (WDTLOCK)	736
9.3.3.1.5	Watchdog Timer Test Register (WDTTEST)	737
9.3.3.2	The Watchdog Module Interrupt Handling Registers	737
9.3.3.2.1	Watchdog Raw Interrupt Status Register (WDTRIS)	737

9.3.3.2.2	Watchdog Masked Interrupt Status Register (WDTMIS)	737
9.3.3.2.3	Watchdog Interrupt Clear Register (WDTICR)	737
9.3.3.2.4	Watchdog Timer Software Reset Register (SRWD)	738
9.3.4	The Watchdog Timer Module Initializations and Configurations	738
9.3.5	Build an Example Watchdog Timer Project	739
9.3.6	The API Functions Used for Watchdog Timer Modules	739
9.3.6.1	The API Functions Used to Configure and Control the Watchdog Timers	740
9.3.6.2	The API Functions Used to Handle Interrupts of the Watchdog Timers	742
9.3.6.3	An Implementation Example of Using API Functions to Control the Watchdog Timer	743
9.4	Universal Serial Bus (USB) Controller	743
9.4.1	The Hardware Configuration of the USB Devices	744
9.4.2	The USB Components and Operational Sequence	745
9.4.3	The Serial Interface Protocol of the USB Communications	747
9.4.4	The USB Interface Used in the Embedded System	748
9.4.5	The USB in the TM4C123GH6PM MCU System	749
9.4.5.1	USB Working as a Device	749
9.4.5.1.1	IN Transactions as a Device	750
9.4.5.1.2	OUT Transactions as a Device	751
9.4.5.1.3	Other Device Functions	752
9.4.5.2	USB Working as a Host	754
9.4.5.2.1	IN Transactions as a Host	755
9.4.5.2.2	OUT Transactions as a Host	755
9.4.5.2.3	Transactions Scheduling	756
9.4.5.2.4	Other Host Functions	756
9.4.5.3	The OTG Mode	757
9.4.5.3.1	Using OTG to Start a Session	758
9.4.5.3.2	Using OTG to Perform Detecting Activity	759
9.4.5.3.3	Using OTG to Perform Host Negotiation	759
9.4.5.4	The USB Module Functional Block Diagram	759
9.4.5.5	The USB Module Control Signals	760
9.4.6	The USB Registers	761
9.4.6.1	USB Host-Related Registers	762
9.4.6.2	USB Device-Related Registers	763
9.4.6.3	USB Host/Device-Related Registers	764
9.4.6.4	USB FIFO-Related Registers	765
9.4.6.5	USB-Interrupt-Related Registers	771
9.4.7	The USB Initializations and Configurations	774
9.4.7.1	Enable and Clock the USB Controller and Related GPIO Ports and Pins	774
9.4.7.2	USB Control Pins Configurations	774
9.4.7.3	Endpoint Configurations	775
9.4.8	A USB Implementation Example Project	775
9.4.9	The USB API Functions Provided by the TivaWare™ Peripheral Driver Library	780

9.4.9.1	The USBClock and USBMode API Functions	781
9.4.9.2	The USBDev API Functions	781
9.4.9.3	The USBHost API Functions	783
9.4.9.4	The USBEndpoint API Functions	784
9.4.9.5	The USBFIFO API Functions	786
9.4.9.6	The USBInterrupt API Functions	786
9.4.9.7	The USBOTG API Functions	788
9.4.10	Build a USB Implementation Example Project Using the API Functions	788
9.5	Chapter Summary	788
	Homework	790

Chapter 10 ARM® Cortex®-M4 Other Peripherals Programming 805

10.1	Overview and Introduction	805
10.2	The Controller Area Network (CAN)	805
10.2.1	CAN Standard Frame	806
10.2.2	CAN Extended Frame	807
10.2.3	Detecting and Signaling Errors	808
10.2.4	The CAN Functional Block Diagram in the TM4C123GH6PM System	809
10.2.5	The CAN Components and Operational Procedures	810
10.2.5.1	CAN Initialization and Configuration Process	811
10.2.5.2	Transmit Message Objects	812
10.2.5.3	Receive Message Objects	815
10.2.5.4	Handle CAN Module Interrupts	817
10.2.5.5	CAN Module Operational Modes	818
10.2.5.6	CAN Clock and Baud Rate Configuration	819
	10.2.5.6.1 Calculate the Bit Time Parameters and Configure the CANBIT Register	821
10.2.6	The CAN Module Registers	823
10.2.6.1	The CAN Global Control and Status Registers	823
	10.2.6.1.1 The CAN Global Control Register (CANCTL)	823
	10.2.6.1.2 The CAN Global Status Register (CANSTS)	824
	10.2.6.1.3 The CAN Error Counter Register (CANERR)	825
	10.2.6.1.4 The CAN Bit Timing Register (CANBIT)	825
	10.2.6.1.5 The CAN Test Register (CANTST)	826
	10.2.6.1.6 The CAN Baud Rate Prescaler Extension Register (CANBRPE)	826
	10.2.6.1.7 The CAN Interrupt Register (CANINT)	827
10.2.6.2	The CAN Interface 1 Registers	828
	10.2.6.2.1 CAN IF1 Command Request Register (CANIF1CRQ)	828
	10.2.6.2.2 CAN IF1 Command Mask Register (CANIF1CMSK)	828
	10.2.6.2.3 CAN IF1 Message Control Register (CANIF1MCTL)	830
	10.2.6.2.4 CAN IF1 Mask 1 Register (CANIF1MSK1)	831
	10.2.6.2.5 CAN IF1 Mask 2 Register (CANIF1MSK2)	831
	10.2.6.2.6 CAN IF1 Arbitration 1 Register (CANIF1ARB1)	831
	10.2.6.2.7 CAN IF1 Arbitration 2 Register (CANIF1ARB2)	832

10.2.6.2.8	CAN IF1 Data A1, CAN IF1 Data A2, CAN IF1 Data B1, CAN IF1 Data B2 (CANIF1DA1–CANIF1DA2, CANIF1DB1~CANIF1DB2) Registers	832
10.2.6.3	The CAN Message Object Registers	833
10.2.7	The CAN Module Interfacing and External Control Signals	833
10.2.8	The CAN API Functions Provided by TivaWare™ Peripheral Driver Library	834
10.2.8.1	Special Data Structures and Enumerations Used in the CAN Programming	834
10.2.8.2	CAN Module Initialization and Configuration Functions	835
10.2.8.3	CAN Module Message Setting and Processing Functions	836
10.2.8.4	CAN Module Interrupt Configuration and Handle Functions	838
10.2.9	A CAN Module Implementation Example Project	838
10.2.9.1	Build a Simple CAN Self-Test Project	839
10.2.9.2	Build the Header File for the CAN Project CANLoopBack	840
10.2.9.3	Build the Source File for the CAN Project CANLoopBack	841
10.2.9.4	Set Up the Environment to Build and Run the Project	846
10.3	The Quadrature Encoder Interface (QEI)	847
10.3.1	Introduction to Quadrature Encoder	847
10.3.2	The Working Principle of the Increment Rotary Encoder	849
10.3.3	The Increment Rotary Encoder Applied in the Closed-Loop Control System	850
10.3.4	The Increment Rotary Encoder Applied in the TM4C123GH6PM MCU System	851
10.3.5	The QEI Module Registers	852
10.3.5.1	QEI Control and Status Registers	852
10.3.5.2	QEI Position Control Registers	854
10.3.5.3	QEI Velocity Control Registers	855
10.3.5.4	QEI Interrupt Processing Registers	855
10.3.6	The QEI Interfacing Signals and Related GPIO Pins	856
10.3.7	The QEI Initialization and Configuration Process	856
10.3.8	QEI API Functions Provided by the TivaWare™ Peripheral Driver Library	857
10.3.8.1	QEI Configuration and Enable API Functions	858
10.3.8.2	QEI Position Capture API Functions	858
10.3.8.3	QEI Velocity Capture API Functions	859
10.3.8.4	QEI Interrupt Handling API Functions	859
10.3.9	An Implementation of Using Rotary Encoder for a Closed-Loop Control System	860
10.3.9.1	Calibration of the Rotary Encoder	861
10.3.9.2	Build the Floating Chart for the Motor Closed-Loop Control System	867
10.3.9.3	Build the Closed-Loop Control Program Based on the Floating Chart	869
10.4	The Continuous and Discrete PID Closed-Loop Control System	871
10.4.1	Identify the Dynamic Model for the Motor Plant	873
10.4.1.1	Format the Input and Output Data for the DC Motor	874

10.4.1.2	Identify the DC Motor Dynamic Model with Identification Toolbox™	876	
10.4.2	Design the PID Controller Using the MATLAB® Control System Toolbox™	878	
10.4.3	Simulate the PID Control System Using the MATLAB® SIMULINK®		881
10.4.4	Build the Control Software to Implement the PID Controller	883	
10.5	The Fuzzy Logic Closed-Loop Control System	887	
10.5.1	The Fuzzification Process	887	
10.5.2	Design of Control Rules	889	
10.5.3	The Defuzzification Process	889	
10.5.4	Apply the Fuzzy Logic Controller to the DC Motor Control System		891
10.5.5	Build the Fuzzy Logic Control Project Fuzzy-Control	894	
10.5.5.1	Create the Header File Fuzzy-Control.h	894	
10.5.5.2	Create the C Source File Fuzzy-Control.c	895	
10.5.5.3	Set Up Environments to Build and Run the Project	898	
10.6	The Analog Comparators	899	
10.6.1	The Analog Comparator Architecture and Functional Block Diagram		899
10.6.2	The Control Registers Used in the Analog Comparator Modules	899	
10.6.3	The Voltage Reference Registers Used in the Analog Comparator Modules	900	
10.6.4	The Interrupt Processing Registers Used in the Analog Comparator Modules	903	
10.6.5	The Input and Output Control Signals Used in the Analog Comparators		903
10.6.6	The Initialization and Configuration Process for the Analog Comparator		904
10.6.7	Build a Project to Test the Functions of the Analog Comparator Module		904
10.6.8	Set Up the Environments to Build and Run the Project	907	
10.7	Chapter Summary	908	
	Homework	909	
Chapter 11 ARM® Floating Point Unit (FPU)			927
11.1	Overview and Introduction	927	
11.2	Three Types of the Floating-Point Data	928	
11.2.1	The Half-Precision Floating-Point Data	928	
11.2.2	The Single-Precision Floating-Point Data	930	
11.2.3	The Double-Precision Floating-Point Data	932	
11.3	The FPU in the Cortex®-M4 MCU	934	
11.3.1	The Architecture of the Floating-Point Registers	934	
11.3.2	The FPU Operational Modes	937	
11.4	Implementing the Floating-Point Unit	938	
11.4.1	Floating-Point Support in CMSIS-Core	938	
11.4.2	Floating-Point Programming in the TM4C123GH6PM MCU System		939
11.4.2.1	FPU in the Direct Register Access Model	940	
11.4.2.2	FPU in the Software Driver Model	942	
11.4.3	An FPU Example Project Using the Direct Register Access Model		942
11.5	Chapter Summary	946	
	Homework	946	

Chapter 12 ARM® Memory Protection Unit (MPU) 951

12.1	Overview and Introduction	951
12.2	Implementation of the MPU	952
12.2.1	Memory Regions, Types, and Attributes	953
12.2.2	MPU Configuration and Control Registers	953
12.2.2.1	The MPU Type Register (MPUTYPE)	954
12.2.2.2	The MPU Control Register (MPUCTRL)	954
12.2.2.3	The MPU Region Number Register (MPUNUMBER)	956
12.2.2.4	MPU Region Base Address Register (MPUBASE)	956
12.2.2.5	The MPU Region Attribute and Size Register (MPUATTR)	957
12.3	Initialization and Configuration of the MPU	959
12.4	Building A Practical Example MPU Project	960
12.4.1	Create a New DRA Model MPU Project DRAMPU	960
12.4.2	Set Up the Environment to Build and Run the Project	963
12.5	The API Functions Provided by the TivaWare™ Peripheral Driver Library	964
12.5.1	The MPU Set Up and Status API Functions	965
12.5.1.1	The API Function MPURegionSet()	966
12.5.2	The MPU Enable and Disable API Functions	967
12.5.3	The MPU Interrupt Handler Control API Functions	968
12.6	Chapter Summary	969
	Homework	970

Index 975**About the Author 987**

Preface

The ARM® Cortex®-M4 MCU is one of the most popular and updated microcontrollers widely implemented in education, industrial, and manufacturing fields in recent years. Because of their relatively simple structure and powerful functions, the ARM® Cortex®-M4 MCU systems have been applied in more and more applications in our real world, including automatic controls, intelligent controls, industrial controls, and academic implementations.

The advantages of using an ARM® Cortex®-M4 microcontroller include but are not limited to the following:

- The ARM® Cortex®-M4 MCU is a 32-bit microcontroller, and it can work independently as a single controller to provide real-time and multifunction controls to effectively and easily control most real objectives in our world.
- The internal bus system used in Cortex®-M4 MCU is 32-bit, and it is based on the so-called Advanced Microcontroller Bus Architecture (AMBA) standard. The AMBA standard provides efficient operations and low power cost on the hardware.
- The main bus interface between the MCU and external components is the Advanced High-performance Bus (AHB), which provide interfaces for memory and system bus, as well as peripheral devices.
- A Nested Vectored Interrupt Controller (NVIC) is used to provide all supports and managements to the interrupt responses and processing to all components in the system.
- The Cortex®-M4 MCU also provides standard and extensive debug features and helps to enable users to easily check and trace their program with breakpoints and steps.
- The TM4C123GXL EVB provides fundamental and basic peripherals and interfaces to enable users to conveniently communicate to other parallel or serial peripherals via GPIO Ports to perform specific control tasks and functions.
- The EduBASE ARM® Trainer provides the most popular I/O devices, such as 4-LED, a 4-bit DIP switch, four 7-segment LEDs, a 4 × 4 keypad working as an input keyboard, a 16 × 2 LCD connected to the LCD Controller HD44780 to work as an output displaying device, two H-Bridge motor drivers, three analog input sensors, a CAN protocol and other peripheral interfaces. All of these I/O devices and interfaces provide great flexibility to enable users to design and build advanced and professional control units applied in our real world.
- The integrated development environment Keil® ARM-MDK μVersion®5 provides an integrated development environment to enable users to easily create, compile, build, and run professional application projects to control and coordinate the entire control system to perform a desired task in short period of time.

The author of this book tries to provide a complete package to cover all components and materials related to ARM® Cortex®-M4 microcontroller systems, including hardware

xxx Preface

and software as well as practical application notes with real examples. All example projects in the book have been compiled, built, and tested. To help students to master the main techniques and ideas, five appendices are provided to facilitate the students to overcome some possible learning curves.

Any questions or comments regarding this book are welcome.

Charlotte, North Carolina

YING BAI

Acknowledgments

The first and most special thanks go to my wife, Yan Wang; I could not finish this book without her sincere encouragements and supports.

Many thanks should be given to the editor Mary Hatcher, who made this book available to the public. You could not find this book on the market without her deep perspective and hard work. The same thanks are extended to the editing team for this book. Without their contributions, it would have been impossible for this book to have been published.

Acknowledgments should also be extended to the following book reviewers for their valuable opinions to the book:

- Dr. Jiang (Linda) Xie, Professor, Department of Electrical and Computer Engineering, University of North Carolina at Charlotte
- Dr. Xiaohong Yuan, Associate Professor, Department of Computer Science, North Carolina A&T State University
- Dr. Daoxi Xiu, Application Analyst Programmer, North Carolina Administrative Office of the Courts
- Dr. Dali Wang, Associate Professor, Department of Physics and Computer Science, Christopher Newport University

Last but not least, thanks should be given to all the people who have supported me while writing this book.

YING BAI

Trademarks and Copyrights

- ARM[®], Cortex[®], Keil[®] and μ Vision[®] are registered trademarks of ARM Limited (or its subsidiaries) in the EU and/or elsewhere.
- ARM7[™], ARM9[™] and ULINK[™] are trademarks of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved.
- MATLAB[®] is a trademark and product of The MathWorks, Inc.
- MATLAB Simulink[®] is a trademark and product of The MathWorks, Inc.
- MATLAB System Identification Toolbox[™] is a trademark and product of The MathWorks, Inc.
- MATLAB System Control Toolbox[™] is a trademark and product of The MathWorks, Inc.
- MATLAB Fuzzy Logic Toolbox[™] is a trademark and product of The MathWorks, Inc.
- Texas Instruments[™] is a trade mark of the Texas Instruments Incorporated.
- Tiva[™] is a trademark and product of the Texas Instruments Incorporated.
- TivaWare[™] is a trademark and product of the Texas Instruments Incorporated.
- LaunchPad[™] is a trademark and product of the Texas Instruments Incorporated.
- Code Composer Studio[™] is a trademark and product of the Texas Instruments Incorporated.
- Stellaris[®] is a trademark and product of the Texas Instruments Incorporated.

Copyright Permissions

All copyright permitted Figures and Tables used in this book are listed below based on the different venders and companies.

Table I lists the copyright permitted Figures and Tables originated by the Texas Instruments Incorporated and used in this book. All those Figures and Tables have been permitted to be re-printed in this book under the copyright permissions of the Texas Instruments Incorporated.

Table I. Copyright Permitted Figures and Tables by Texas Instruments Incorporated.

Chapter	Figures and Tables	Page
Chapter 2	Figure 2.14: Block diagram of TM4C123GH6PM MCU.	39
	Figure 2.17: Function block diagram of Analog/Digital GPIO control.	48
	Figure 2.20: The TM4C123GXL evaluation board.	73
	Figure 2.21: The functional block diagram of the LaunchPad board.	76
	Table 2.3: Exception and interrupt types and priority numbers.	36
	Table 2.8: GPIO Pins with special considerations.	58
Chapter 4	Figure 4.34: An example of using mask byte to do data reading and writing.	230
Chapter 5	Figure 5.6: An example of the NVIC Priority Level Register PRI0.	276
Chapter 6	Figure 6.3: Bit field and function on BOOTCFG Register.	339
	Figure 6.4: Bit field and function on FMA Register.	341
	Figure 6.5: Bit field and function on the FMC Register.	342
	Figure 6.6: Bit field and function on the FMC2 Register.	343
	Figure 6.7: Bit field and function on FCRIS Register.	344
	Figure 6.8: Bit field and function on FCIM Register.	346
	Figure 6.9: Bit field and function on FCMISC Register.	347
	Figure 6.11: The functional block diagram of the EEPROM.	355
	Figure 6.12: The bit field values and related functions in the EEDONE register.	358
Chapter 7	Figure 6.13: The bit field values and related functions in the EEPROT register.	360
	Figure 7.12: The bit fields for the ADCSSMUX0 register.	452
	Figure 7.13: The bit fields for the ADCSSMUX1, 2 registers.	453
	Figure 7.14: The bit fields for the ADCSSMUX3 register.	454
	Figure 7.15: The bit fields for the ADCSSCTL0 register.	455
	Figure 7.16: The bit fields for the ADCSSCTL1, 2 registers.	456
	Figure 7.17: The bit fields for the ADCSSCTL3 register.	458
	Figure 7.32: PWM Count-Down and Count Up/Down modes.	488
	Figure 7.33: An example of using the count-up/down mode to generate PWM outputs.	489
	Figure 7.34: The output PWM signals generated by Dead-Band generator.	490

Table I (Continued)

Chapter	Figures and Tables	Page
	Figure 7.35: The detailed block diagram for the PWM Generator block.	491
	Figure 7.36: Bit fields in the PWM generator control register.	492
	Figure 7.37: Bit fields in the PWM generator A register.	494
	Figure 7.41: Architecture and functional block diagram of PWM module.	503
	Figure 7.42: Bit fields in the Run-Mode Clock Configuration (RCC) register.	504
Chapter 8	Figure 8.4: Functional block diagram of the SSI module.	556
	Figure 8.5: Operational timing sequence of the TI synchronous serial frame.	558
	Figure 8.6: The operational sequence for Freescale SPI frame (SPO = SPH = 0).	559
	Figure 8.7: The operational sequence for Freescale SPI frame (SPO = 0, SPH = 1).	559
	Figure 8.8: The operational sequence for MACROWIRE frame.	560
	Figure 8.11: The bit field and functions of the SSICR0 Register.	563
	Figure 8.42: The I2C bus configuration and status.	612
	Figure 8.43: The definition of START and STOP conditions.	612
	Figure 8.44: The functional block diagram of each I2C module.	613
	Figure 8.45: The I2C data transfer format and frame.	614
	Figure 8.46: The operational sequence of the master working in the transmit mode.	615
	Figure 8.47: The operational sequence of the master working in the receive mode.	617
	Figure 8.48: The operational sequence of the I2C module working in the slave mode.	618
	Figure 8.62: The functional block diagram for one UART module.	644
	Figure 8.65: Bit configurations of the UARTCTL register.	652
Chapter 9	Figure 9.1: The architecture and block diagram of one GPTM block.	693
	Figure 9.3: An example of using a count-down timer to detect input edge events.	700
	Figure 9.4: An example of using the count-down mode timer to detect the edge time.	701
	Figure 9.5: An example of using Timer A to generate a PWM signal.	703
	Figure 9.20: The functional block diagram of the watchdog modules.	734
	Figure 9.27: Functional block diagram of the USB module.	760
Chapter 10	Figure 10.2: A typical standard CAN frame format.	807
	Figure 10.3: Functional block diagram of the CAN modules.	809
	Figure 10.5: A normal CAN bit time configuration.	820
	Figure 10.25: Functional block diagram of the QEI Modules.	851
	Figure 10.26: The inversion and swapping logic circuit.	852
	Figure 10.63: Architecture and functional block diagram of the Analog Comparator modules.	900

Table II lists the copyright permitted Figures and Tables originated by the ARM Limited and used in this book. All those Figures and Tables have been reproduced with permission from ARM Limited. Copyright © ARM Limited.

Table II. Copyright Permitted Figures and Tables by ARM Limited.

Chapter	Figures and Tables	Page
Chapter 3	Figure 3.4: The components included in the MDK Core.	
	Figure 3.5: Components in the Software Packs.	
	Figure 3.9: The opened Keil® MDK-ARM µVersion® 5.1 suite.	87
	Figure 3.11: The opened Hello World project.	88
	Figure 3.12: The opened source file hello.c.	92
	Figure 3.13: Rebuild the hello project.	93
	Figure 3.14: The building result of the hello project.	94
	Figure 3.15: The debugging result of the hello world project.	95
	Figure 3.16: The opened MDK-ARM IDE.	95
	Figure 3.17: The opened Device Database wizard.	96
	Figure 3.18: The detailed information for the device TM4C123GH6PM.	97
	Figure 3.19: The License Management wizard.	98
	Figure 3.20: An example of the Configuration submenu.	99
	Figure 3.22: The Select Device for Target wizard.	99
	Figure 3.23: The Manage Run-Time Environment wizard.	100
	Figure 3.24: The selected components in the Manage Run-Time Environment wizard.	103
	Figure 3.25: The new project wizard.	
	Figure 3.26: The Add New Item to Group wizard.	104
	Figure 3.28: The finished codes for the source file MyProject.c.	105
	Figure 3.29: Add a header file MyProject.h into the project.	106
	Figure 3.31: The finished header file MyProject.h.	108
	Figure 3.32: The project building process.	108
	Figure 3.33: The finished debugger checking wizard.	109
	Figure 3.34: The download process for our project.	110
	Figure 3.35: The debug process for our project.	111
	Figure 3.36: The Components, Environment, Books wizard.	111
	Figure 3.37: The Manage Run-Time Environment wizard for our sample project.	112
	Figure 3.38: Functions provided by the Options for Target Project wizard.	114
	Figure 3.39: The Target option for the sample project MyProject.	114
	Figure 3.40: The Output option for the sample project MyProject.	116
	Figure 3.41: The Listing option for the sample project MyProject.	116
	Figure 3.42: The Debug option for the sample project MyProject.	117
	Figure 3.43: The Utilities option for the sample project MyProject.	117
	Figure 3.44: The opened Settings wizard.	119
	Figure 3.45: An example of using System Viewer for TIMER0 device.	119
	Figure 3.46: The Nested Vectored Interrupt Controller configuration dialog.	120
	Figure 3.50: A debug example for our sample project MyProject.	123
	Figure 3.51: The optimization wizard for our sample project.	125
	Figure 3.52: An example of using the code optimization under the Target tab.	134
		139
		141
Chapter 4	Figure 4.38: The finished Options wizard.	236
Chapter 6	Figure 6.26: The running result of the project DRAFlash.	387
	Figure 6.27: The 1-KB erased flash memory block (0x1000~0x13FF).	388

Table II (Continued)

Chapter	Figures and Tables	Page
	Figure 6.32: The running result of the DRAFlashInt project.	397
	Figure 6.34: The running result of the project DRAFlashBuffer.	400
	Figure 6.38: The 18 read-out data stored in the data array prData[].	414
	Figure 6.42: The read back data stored in the prData[] array.	420
Chapter 8	Figure 8.74: Running result of the project DRAUART.	664
	Figure 8.75: Running result of the project Lab8_5.	685
	Figure 8.76: Running result of the project Lab8_6.	689
Chapter 11	Figure 11.6: The FPU is automatically used in MDK-ARM μ Version 5 IDE.	939
	Figure 11.11: The running result of the project DRAFPU.	945

Table III list the copyright permitted Figures and Tables originated by the MathWorks, Inc. and used in this book. All those Figures and Tables have been permitted to be re-printed in this book under the copyright permissions of the MathWorks, Inc.

Chapter	Figures and Tables	Page
	Figure 10.38: The MATLAB Script file getMDData.m.	875
	Figure 10.39: Load the data array mdata.dat into the MATLAB Workspace.	876
	Figure 10.40: The opened Identification Toolbox and Import data wizard.	876
	Figure 10.41: The modified data array mdatad in the identification Toolbox.	877
	Figure 10.42: The opened Process Models wizard.	878
	Figure 10.43: The identified model responses and analysis.	879
Chapter 10	Figure 10.44: Commands used to set transfer function of the DC motor and start PID tuner.	880
	Figure 10.45: The opened PID Tuner and the tuning result.	880
	Figure 10.46: The opened SIMULINK window.	881
	Figure 10.47: The finished SIMULINK block connections.	882
	Figure 10.48: The simulated step response result.	883
	Figure 10.51: The step response of the actual closed-loop motor control system.	886
	Figure 10.55: Graphic representation of the control rules.	890
	Figure 10.56: The fuzzy output surface or envelope.	890
	Figure 10.62: The step response of the fuzzy logic control system.	898
Chapter 11	Figure 11.12: The plotting result for the data array gSDData[].	945

About the Companion Website

This book is accompanied by a companion website: <http://www.wiley.com/go/armbai>
The website includes:

- Class Projects
- Appendix A
- Appendix B
- Appendix C
- Appendix D
- Appendix E

If you are an instructor and adopted this book for your course, please email ieeeproposals@wiley.com to get access to the instructor files for this book.