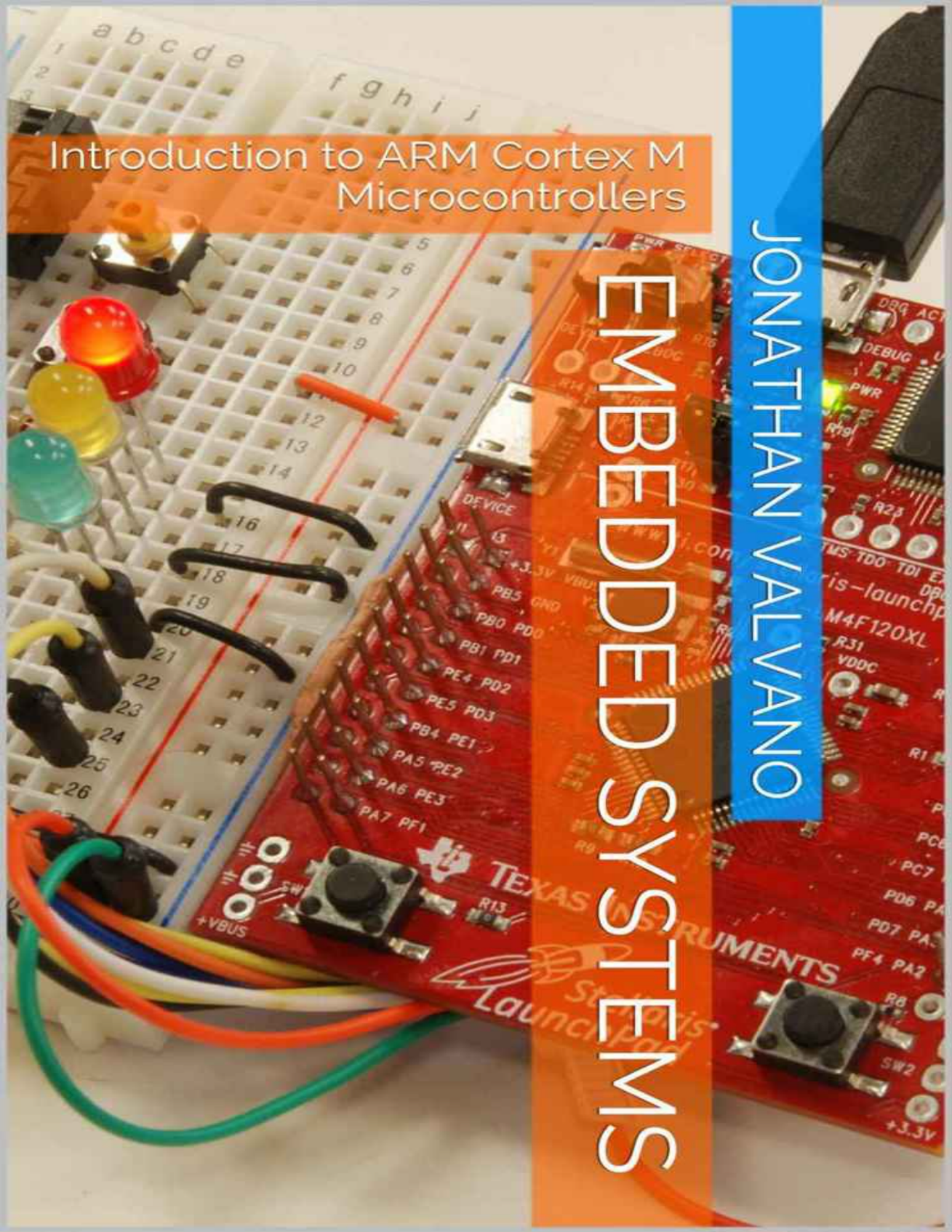


Introduction to ARM Cortex M
Microcontrollers

JONATHAN VALVANO

EMBEDDED SYSTEMS



EMBEDDED SYSTEMS:

INTRODUCTION TO ARM®CORTEX[®] -M MICROCONTROLLERS

Volume 1
Fifth Edition
June 2014

Jonathan W. Valvano

Fifth edition
2nd printing
June 2014

ARM and uVision are registered trademarks of ARM Limited.
Cortex and Keil are trademarks of ARM Limited.
Stellaris and Tiva are registered trademarks Texas Instruments.
Code Composer Studio is a trademark of Texas Instruments.
All other product or service names mentioned herein are the trademarks of their respective owners.

In order to reduce costs, this college textbook has been self-published. For more information about my classes, my research, and my books, see <http://users.ece.utexas.edu/~valvano/>

For corrections and comments, please contact me at: valvano@mail.utexas.edu. Please cite this book as: J. W. Valvano, Embedded Systems: Introduction to ARM® Cortex □ -M Microcontrollers, Volume 1, <http://users.ece.utexas.edu/~valvano/>, ISBN: 978-1477508992.

Copyright © 2014 Jonathan W. Valvano

All rights reserved. No part of this work covered by the copyright herein may be reproduced, transmitted, stored, or used in any form or by any means graphic, electronic, or mechanical, including but not limited to photocopying, recording, scanning, digitizing, taping, web distribution, information networks, or information storage and retrieval, except as permitted under Section 107 or 108 of the 1976 United States Copyright Act, without the prior written permission of the publisher.

ISBN-13: 978-1477508992

ISBN-10: 1477508996

Table of Contents

[Preface to the Fifth Edition](#)

[Preface](#)

[Acknowledgements](#)

[1. Introduction to Computers and Electronics](#)

[1.1. Review of Electronics](#)

[1.2. Binary Information Implemented with MOS transistors](#)

[1.3. Digital Logic](#)

[1.4. Digital Information stored in Memory](#)

[1.5. Numbers](#)

[1.6. Character information](#)

[1.7. Computer Architecture](#)

[1.8. Flowcharts and Structured Programming](#)

[1.9. Concurrent and Parallel Programming](#)

[1.10. Exercises](#)

[2. Introduction to Embedded Systems](#)

[2.1. Embedded Systems](#)

[2.2. Applications Involving Embedded Systems](#)

[2.3. Product Life Cycle](#)

[2.4. Successive Refinement](#)

[2.5. Quality Design](#)

[2.5.1. Quantitative Performance Measurements](#)

[2.5.2. Qualitative Performance Measurements](#)

[2.5.3. Attitude](#)

[2.6. Debugging Theory](#)

[2.7. Switch and LED Interfaces](#)

[2.8. Introduction to C](#)

[2.9. Exercises](#)

[3. Introduction to the ARM \$\square\$ Cortex \$\square\$ -M Processor](#)

[3.1. Cortex \$\square\$ -M Architecture](#)

[3.1.1. Registers](#)

[3.1.2. Reset](#)

[3.1.3. Memory](#)

[3.1.4. Operating Modes](#)

[3.2. The Software Development Process](#)

[3.3. ARM Cortex-M Assembly Language](#)

[3.3.1. Syntax](#)

[3.3.2. Addressing Modes and Operands](#)

[3.3.3. Memory Access Instructions](#)

[3.3.4. Logical Operations](#)

[3.3.5. Shift Operations](#)

[3.3.6. Arithmetic Operations](#)

[3.3.7. Stack](#)

[3.3.8. Functions and Control Flow](#)

[3.3.9. Assembler Directives](#)

[3.3.10. First Example Project](#)

[3.4. Simplified Machine Language Execution](#)

[3.5. CISC versus RISC](#)

[3.6. Details Not Covered in this Book](#)

[3.7. Exercises](#)

[4. Introduction to Input/Output](#)

[4.1. Texas Instruments Microcontroller I/O pins](#)

[4.1.1. Texas Instruments LM3S1968 I/O pins](#)

[4.1.2. Texas Instruments TM4C123 LaunchPad I/O pins](#)

[4.1.3. Texas Instruments TM4C1294 Connected LaunchPad I/O pins](#)

[4.2. Basic Concepts of Input and Output Ports](#)

[4.2.1. I/O Programming and the Direction Register](#)

[4.2.2. Switch Inputs and LED Outputs](#)

[4.3. Phase-Lock-Loop](#)

[4.4. SysTick Timer](#)

[4.5. Standard I/O Driver and the printf Function](#)

[4.6. Debugging monitor using an LED](#)

[4.7. Performance Debugging](#)

[4.7.1. Instrumentation](#)

[4.7.2. Measurement of Dynamic Efficiency](#)

[4.8. Exercises](#)

[4.9. Lab Assignments](#)

[5. Modular Programming](#)

[5.1. C Keywords and Punctuation](#)

[5.2. Modular Design using Abstraction](#)

[5.2.1. Definition and Goals](#)

[5.2.2. Functions, Procedures, Methods, and Subroutines](#)

[5.2.3. Dividing a Software Task into Modules](#)

[5.2.4. How to Draw a Call Graph](#)

[5.2.5. How to Draw a Data Flow Graph](#)

[5.2.6. Top-down versus Bottom-up Design](#)

[5.3. Making Decisions](#)

5.3.1. Conditional Branch Instructions

5.3.2. Conditional if-then Statements

5.3.3. switch Statements

5.3.4. While Loops

5.3.5. Do-while Loops

5.3.6. For Loops

5.4. *Assembly Macros

5.5. *Recursion

5.6. Writing Quality Software

5.6.1. Style Guidelines

5.6.2. Comments

5.6.3. Inappropriate I/O and Portability

5.7. How Assemblers Work

5.8. Functional debugging

5.8.1. Stabilization

5.8.2. Single Stepping

5.8.3. Breakpoints with Filtering

5.8.4. Instrumentation: Print Statements

5.8.5. Desk checking

5.9. Exercises

5.10. Lab Assignments

6. Pointers and Data Structures

6.1. Indexed Addressing and Pointers

6.2. Arrays

6.3. Strings

6.4. Structures

6.5. Finite State Machines with Linked Structures

[6.5.1. Abstraction](#)

[6.5.2. Moore Finite State Machines](#)

[6.5.3. Mealy Finite State Machines](#)

[6.6. *Dynamically Allocated Data Structures](#)

[6.6.1. *Fixed Block Memory Manager](#)

[6.6.2. *Linked List FIFO](#)

[6.7. Matrices and Graphics](#)

[6.8. *Tables](#)

[6.9. Functional Debugging](#)

[6.9.1. Instrumentation: Dump into Array without Filtering](#)

[6.9.2. Instrumentation: Dump into Array with Filtering.](#)

[6.10. Exercises](#)

[6.11. Lab Assignments](#)

[7. Variables, Numbers, and Parameter Passing](#)

[7.1. Local versus global](#)

[7.2. Stack rules](#)

[7.3. Local variables allocated on the stack](#)

[7.4. Stack frames](#)

[7.5. Parameter Passing](#)

[7.5.1. Parameter Passing in C](#)

[7.5.2. Parameter Passing in Assembly Language](#)

[7.5.3. C Compiler Implementation of Local and Global Variables](#)

[7.6. Fixed-point Numbers](#)

[7.7. Conversions](#)

[7.8. *IEEE Floating-point numbers](#)

[7.9. Exercises](#)

[7.10. Lab Assignments](#)

8. Serial and Parallel Port Interfacing

8.1. General Introduction to Interfacing

8.2. Universal Asynchronous Receiver Transmitter (UART)

8.2.1. Asynchronous Communication

8.2.2. LM3S/TM4C UART Details

8.2.3. UART Device Driver

8.3. Synchronous Serial Interface, SSI

8.4. Nokia 5110 Graphics LCD Interface

8.5. Scanned Keyboards

8.6. Binary actuators

8.6.1. Interface

8.6.2. Electromagnetic and Solid State Relays

8.6.3. Solenoids

8.7. *Pulse-width modulation

8.8. *Stepper motors

8.9. Exercises

8.10. Lab Assignments

9. Interrupt Programming and Real-time Systems

9.1. I/O Synchronization

9.2. Interrupt Concepts

9.3. Interthread Communication and Synchronization

9.4. NVIC on the ARM Cortex-M Processor

9.5. Edge-triggered Interrupts

9.6. SysTick Periodic Interrupts

9.7. Timer Periodic Interrupts

9.8. Hardware debugging tools

9.9. Profiling

9.9.1 Profiling using a software dump to study execution pattern

9.9.2. Profiling using an Output Port

9.9.3. *Thread Profile

9.10. Exercises

9.11. Lab Assignments

10. Analog I/O Interfacing

10.1. Approximating continuous signals in the digital domain

10.2. Digital to Analog Conversion

10.3. Music Generation

10.4. Analog to Digital Conversion

10.4.1. LM3S/TM4C ADC details

10.4.2. ADC Resolution

10.5. Real-time data acquisition

10.6. Exercises

10.7. Lab Assignments

11. Communication Systems

11.1. Introduction

11.2. Reentrant Programming and Critical Sections

11.3. Producer-Consumer using a FIFO Queue

11.3.1. Basic Principles of the FIFO Queue

11.3.2. FIFO Queue Analysis

11.3.3. FIFO Queue Implementation

11.3.4. Double Buffer

11.4. Serial port interface using interrupt synchronization

11.5. *Distributed Systems.