

Embedded Systems Design

Using the MSP430FR2355 LaunchPad™

SECOND EDITION

Brock J. LaMeres

 Springer

**EMBEDDED SYSTEMS DESIGN USING
THE MSP430FR2355
LAUNCHPAD™**

**EMBEDDED SYSTEMS DESIGN USING
THE MSP430FR2355
LAUNCHPAD™**

2ND EDITION

Brock J. LaMeres

 **Springer**

Brock J. LaMeres
Department of Electrical and Computer Engineering
Montana State University
Bozeman, MT, USA

ISBN 978-3-031-20887-4 ISBN 978-3-031-20888-1 (eBook)
<https://doi.org/10.1007/978-3-031-20888-1>

© The Editor(s) (if applicable) and The Author(s), under exclusive license to
Springer Nature Switzerland AG 2020, 2023

This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

Why Another Book on Embedded Systems?

Embedded computers represent one of the most pervasive technologies of our time. When most people think of computers, they think of their laptops and workstations, or even the servers that are the backbone of the Internet; however, when one begins to contemplate how many technologies around them use small, inexpensive computers embedded as their “brains,” one begins to more fully understand the magnitude of how many embedded computers exist in our society. If you look around any room you will see technology that is so commonplace, you may not even realize that most of it is run by an internal computer. Technologies such as appliances, thermostats, handheld devices, projectors, televisions, personal assistants, and radios all have small computers within them that control their operation. As one starts to notice all of the embedded computers around them, they will quickly notice that there are orders of magnitude more embedded computers than the standard general-purpose computers that run Windows® or iOS™. As these technologies become *smarter* and are Wi-Fi connected, we move into era called the *Internet of Things*. This next technological paradigm shift will turn all of the embedded computers into a collaborative network in an attempt to make our lives better by automating taking mundane tasks.

Simultaneous to the exponential increase in embedded computers in our society is the need for engineers and software developers that can build and program these systems. Due to the high popularity of embedded computers, there is also a wide range of embedded computer platforms. As such, the textbooks that exist to aid in the education of embedded computers are somewhat fragmented. Textbooks for embedded computers aren’t able to be written at only the theoretical level such as circuits and electronics textbooks. Instead, they must identify a computer platform and focus the content on that specific embedded computer technology. This leads to a large number of embedded systems books to support the large number of embedded platforms. This also leads to books becoming obsolete much faster than traditional engineering textbooks.

One reason for a new book in this area is creating comprehensive content around a widely popular embedded computer, the Texas Instruments MSP430. The MSP430 is a modern computer platform that is mature in the market and has a great deal of technical and educational support behind it. This means that this platform will be relevant and supported for at least the next 10–15 years.

A second reason for a new book in this area is to provide content the way people actual learn (i.e., by doing). Current embedded systems books, even those targeting the MSP430, tend to only provide small code segments that can’t be directly run on a target platform. This new book approaches content delivery around the model that the reader is sitting at a computer with an MSP430 plugged in and running each of the examples as they move through the book. This *learn-a-little, do-a-little* is a proven pedagogical strategy. It also supports both active learning within the classroom and self-regulated learning for individual readers.

A third reason for a new book in this area is to provide a seamless delivery of both assembly language and C language programming of the MSP430. Most embedded systems books are structured in one of three ways: (1) they only cover assembly; (2) they only cover C; (3) they attempt to cover both but don’t provide sufficient details of either. This new book will begin in assembly language as a means to show the lower-level operation of the computer hardware. It will then move into C to implement more complex systems that require abstracting the lower-level hardware. With the design of the book consisting of examples in both languages that can be directly coded and run, there will not be a void between the examples and real implementation of the concepts as seen in other books.

The three guiding principles for the design of this book are:

Learn by example This book is designed to teach the material the way it is learned, through example. Every concept that is covered in this book is supported by numerous programming examples that provide the reader with a step-by-step explanation for how and why the computer is doing what it is doing.

Learn by doing This book targets the Texas Instruments MSP430 microcontroller. This platform is a widely popular, low-cost embedded system that is used to illustrate each concept in the book. The book is designed for a reader who is at their computer with the MSP430 plugged in so that each example can be coded and run as they learn.

Build a foundational understanding first, then move into abstraction This book teaches the basic operation of an embedded computer using assembly language first so that the computer operation can be explored at a lower level. Once complicated systems are introduced (i.e., serial communication and analog-to-digital converters), the book moves into the C programming language. Moving to C allows the learner to abstract the operation of the lower-level hardware and focus on understanding how to “make things work.” By spending time in assembly to understand the underlying hardware, the transition to C can happen more rapidly while still leaving the reader with a foundational understanding of the underlying hardware.

If this learning model is followed, the reader will come away with much more than knowledge of facts and figures; they will have a skillset and experience with embedded systems that will be both marketable and of key importance to our society.

This second edition adds two highly requested chapters: Chaps. 16 and 17. Both chapters provide example programs on how to configure these systems and observe their operation.

How to Use This Book

This book is most effective when the reader has the *Texas Instruments Inc. MSP430FR2355 LaunchPad™ Development Kit* and codes along with the material. All examples can be directly run on the LaunchPad™ board. All programs were created and compiled using the *Texas Instruments Inc. Code Composer Studio (CCS)*. This development environment is free from Texas Instruments and can be run on multiple operating systems. Each of the examples noted with a keyboard are ones that are intended to be coded in CCS and run on the LaunchPad™ board.

There are three supporting documents that should also be downloaded from Texas Instruments to provide background for the material in this book. The first is the *MSP430FR4xx and MSP430FR2xx Family User's Guide*¹. This document gives the general concept of operation for the MSP430 architecture that is used on the specific microcontroller targeted in this book. Throughout this book, this document is referred to as the “MSP430 User’s Guide.” The second document that should be retrieved is the *MSP430FR235x, MSP430FR215x Mixed-Signal Microcontrollers Device-Specific Data Sheet*². This second document provides the specific details of the MSP430FR2355 microcontroller device that is used in each example. Throughout this book, this document is referred to as the “Device-Specific Data Sheet.” The final document that should be retrieved is the *MSP430FR235x LaunchPad™ Development Kit User's Guide*³. This document describes the circuitry implemented on the LaunchPad™ board. This document is critical to understanding the types of input/output and programming capability available on the LaunchPad™ board.

Additional Resources

Supporting videos will be posted to the author's YouTube channel, which can be found at https://www.youtube.com/c/DigitalLogicProgramming_LaMeres. This channel contains many other videos beyond embedded systems to help people learn digital logic, basic programming, and robotics. Subscribe to this channel in order to be notified when new videos for this book are posted.

Bozeman, MT, USA

Brock J. LaMeres

Acknowledgments

Endless thanks to my family for support of this project. JoAnn, Alexis, and Kylie: You are my rocks and rockstars.

A huge thank you to all the Montana State University engineering students that helped with the development of this book. They proofed, made suggestions, helped create examples and labs, created slides, and even drew out explanations for the solutions manual. Go Bobcats!

Contents

1: INTRODUCTION TO EMBEDDED SYSTEMS	1
1.1 WHAT IS AN EMBEDDED SYSTEM?	1
2: DIGITAL LOGIC BASICS	7
2.1 NUMBER SYSTEMS	7
2.1.1 <i>Positional Number Systems</i>	7
2.1.2 <i>Base Conversion</i>	12
2.1.3 <i>Binary Arithmetic</i>	19
2.1.4 <i>Unsigned and Signed Numbers</i>	22
2.2 COMBINATIONAL LOGIC	29
2.2.1 <i>Basic Gates</i>	29
2.2.2 <i>Boolean Algebra</i>	31
2.2.3 <i>Combinational Logic Synthesis</i>	34
2.2.4 <i>MSI Logic</i>	56
2.3 SEQUENTIAL LOGIC	71
2.3.1 <i>Sequential Logic Storage Devices</i>	71
2.3.2 <i>Finite State Machines</i>	86
2.4 MEMORY	104
2.4.1 <i>Memory Terminology</i>	104
2.4.2 <i>Memory Architecture</i>	106
2.4.3 <i>Memory Technologies</i>	110
3: COMPUTER SYSTEMS	121
3.1 COMPUTER OVERVIEW	121
3.2 COMPUTER HARDWARE	122
3.2.1 <i>Program Memory</i>	123
3.2.2 <i>Data Memory</i>	123
3.2.3 <i>Central Processing Unit</i>	123
3.2.4 <i>Input/Output Ports</i>	125
3.2.5 <i>Bus System</i>	125
3.3 COMPUTER SOFTWARE	126
3.3.1 <i>Classes of Instructions</i>	126
3.3.2 <i>Op-codes and Operands</i>	127
3.3.3 <i>Program Development Flow</i>	129
4: THE MSP430	135
4.1 MSP430 HARDWARE OVERVIEW	135
4.1.1 <i>Word Versus Byte Memory Access</i>	136
4.1.2 <i>Program Memory</i>	136
4.1.3 <i>Data Memory</i>	136
4.1.4 <i>Central Processing Unit</i>	136
4.1.5 <i>Input/Output Ports and Peripherals</i>	138

4.1.6	<i>Bus System</i>	140
4.1.7	<i>MSP430 Part Numbering</i>	141
4.2	MSP430 SOFTWARE OVERVIEW	142
4.2.1	<i>The MSP430 Instruction Set</i>	142
4.2.2	<i>Word (.W) Versus Byte (.B) Operations</i>	144
4.2.3	<i>The TI Code Composer Studio Development Environment</i>	145
4.3	MSP430FR2355 LAUNCHPAD™ DEVELOPMENT KIT	146
5:	GETTING STARTED PROGRAMMING THE MSP430 IN ASSEMBLY	153
5.1	THE ANATOMY OF AN ASSEMBLY PROGRAM FILE	153
5.1.1	<i>Instruction Statements</i>	153
5.1.2	<i>Assembler Directives</i>	154
5.1.3	<i>Miscellaneous Syntax Notes</i>	157
5.2	YOUR FIRST PROGRAM: BLINKING LED	158
5.3	USING THE CCS DEBUGGER	162
5.3.1	<i>Resume, Terminate, and Suspend</i>	162
5.3.2	<i>Breakpoints</i>	163
5.3.3	<i>Viewing Register Contents</i>	164
5.3.4	<i>Viewing the Contents of Memory</i>	166
5.3.5	<i>Stepping Your Program</i>	166
6:	DATA MOVEMENT INSTRUCTIONS	171
6.1	THE MOV INSTRUCTION WITH REGISTER MODE (R _N) ADDRESSING	171
6.2	THE MOV INSTRUCTION WITH IMMEDIATE MODE (#N) ADDRESSING	173
6.3	THE MOV INSTRUCTION WITH ABSOLUTE MODE (&ADDR) ADDRESSING	175
6.4	THE MOV INSTRUCTION WITH SYMBOLIC MODE (ADDR) ADDRESSING	177
6.5	THE MOV INSTRUCTION WITH INDIRECT REGISTER MODE (@R _N) ADDRESSING	179
6.6	THE MOV INSTRUCTION WITH INDIRECT AUTOINCREMENT MODE (@R _N +) ADDRESSING	181
6.7	THE MOV INSTRUCTION WITH INDEXED MODE (X(R _N)) ADDRESSING	183
7:	DATA MANIPULATION INSTRUCTIONS	191
7.1	ARITHMETIC INSTRUCTIONS	191
7.1.1	<i>Addition Instructions</i>	191
7.1.2	<i>Subtraction Instructions</i>	195
7.1.3	<i>Increments and Decrements</i>	198
7.2	LOGIC INSTRUCTIONS	199
7.3	BIT SET AND BIT CLEAR INSTRUCTIONS	203
7.4	TEST INSTRUCTIONS	204
7.5	ROTATE OPERATIONS	206
8:	PROGRAM FLOW INSTRUCTIONS	213
8.1	UNCONDITIONAL JUMPS AND BRANCHES	213
8.2	CONDITIONAL JUMPS	215
8.2.1	<i>Carry-Based Jumps</i>	215
8.2.2	<i>Zero-Based Jumps</i>	217
8.2.3	<i>Negative-Based Jumps</i>	218

8.2.4	<i>Overflow-Based Jumps</i>	219
8.3	IMPLEMENTING COMMON PROGRAMMING CONSTRUCTS IN ASSEMBLY	220
8.3.1	<i>Implementing While() Loop Functionality</i>	220
8.3.2	<i>Implementing For() Loop Functionality</i>	221
8.3.3	<i>Implementing If/Else Functionality</i>	223
8.3.4	<i>Implementing Switch/Case Functionality in Assembly</i>	224
8.4	FLOW CHARTS	225
9:	DIGITAL I/O	231
9.1	THE MSP430 DIGITAL I/O SYSTEM	231
9.1.1	<i>Port Direction Registers (PxDIR)</i>	233
9.1.2	<i>Port Input Registers (PxIN)</i>	233
9.1.3	<i>Port Output Registers (PxOUT)</i>	233
9.1.4	<i>Port Pull-up or Pull-down Resistor Enable Registers (PxREN)</i>	233
9.1.5	<i>Port Function Select Registers (PxSEL1 and PxSEL0)</i>	235
9.1.6	<i>Digital I/O Enabling After Reset</i>	235
9.1.7	<i>Using Literal Definitions from the MSP430.H Header File</i>	236
9.2	DIGITAL OUTPUT PROGRAMMING	237
9.3	DIGITAL INPUT PROGRAMMING	240
10:	THE STACK AND SUBROUTINES	247
10.1	THE STACK	247
10.2	SUBROUTINES	251
11:	INTRODUCTION TO INTERRUPTS	255
11.1	THE CONCEPT OF AN INTERRUPT	255
11.1.1	<i>Interrupt Flags (IFG)</i>	255
11.1.2	<i>Interrupt Priority and Enabling</i>	257
11.1.3	<i>Interrupt Vectors</i>	258
11.1.4	<i>Operation of the Stack during an IRQ</i>	260
11.1.5	<i>Interrupt Service Routines (ISR)</i>	261
11.1.6	<i>Nested Interrupts</i>	261
11.1.7	<i>Interrupt Servicing Summary</i>	262
11.1.8	<i>MSP430FR2355 Interrupts</i>	263
11.2	MSP430FR2355 PORT INTERRUPTS	265
12:	INTRODUCTION TO TIMERS	273
12.1	TIMER OVERVIEW	273
12.2	TIMER OVERFLOWS ON THE MSP430FR2355	278
12.3	TIMER COMPARES ON THE MSP430FR2355	289
12.4	CREATING PULSE WIDTH MODULATED SIGNALS USING TIMER COMPARES	293
12.5	TIMER CAPTURES ON THE MSP430FR2355	297
13:	SWITCHING TO THE C LANGUAGE	301
13.1	BASICS OF C PROGRAMMING ON THE MSP430	301
13.1.1	<i>While() Loops in C</i>	302

13.1.2	<i>For()</i> Loops in C	304
13.1.3	<i>If/Else</i> Statements in C	305
13.1.4	<i>Switch/Case</i> Statements in C	306
13.1.5	<i>Arithmetic Operators</i> in C	307
13.1.6	<i>Bitwise Logic Operators</i> in C	307
13.2	DIGITAL I/O IN C	310
13.3	INTERRUPTS IN C	313
13.4	TIMERS IN C	316
14:	SERIAL COMMUNICATION IN C	327
14.1	UNIVERSAL ASYNCHRONOUS RECEIVER/TRANSMITTER (UART)	327
14.1.1	<i>The UART Standard</i>	327
14.1.2	<i>UART Transmit on the MSP430FR2355</i>	333
14.1.3	<i>UART Receive on the MSP430FR2355</i>	355
14.2	SERIAL PERIPHERAL INTERFACE (SPI)	359
14.2.1	<i>The SPI Protocol</i>	359
14.2.2	<i>SPI Master Operation on the MSP430FR2355</i>	363
14.2.3	<i>SPI Slave Operation on the MSP430FR2355</i>	384
14.3	INTER-INTEGRATED CIRCUIT (I2C) BUS	385
14.3.1	<i>The I2C Protocol</i>	385
14.3.2	<i>I2C Master Operation on the MSP430FR2355</i>	395
14.3.3	<i>I2C Slave Operation on the MSP430FR2355</i>	417
15:	ANALOG-TO-DIGITAL CONVERTERS	423
15.1	ANALOG-TO-DIGITAL CONVERTERS	423
15.2	ADC OPERATION ON THE MSP430FR2355	426
16:	THE CLOCK SYSTEM	439
16.1	OVERVIEW OF THE MSP430FR2355 CLOCK SYSTEM	439
16.1.1	<i>Internal Very Low-Power Low-Frequency Oscillator (VLO)</i>	439
16.1.2	<i>Internal Trimmed Low-Frequency Reference Oscillator (REFO)</i>	439
16.1.3	<i>External XT1 Oscillator (XT1)</i>	439
16.1.4	<i>Internal Digitally Controlled Oscillator (DCO)</i>	440
16.1.5	<i>Internal High-Frequency Oscillator (MODCLK)</i>	443
16.1.6	<i>Master Clock (MCLK)</i>	443
16.1.7	<i>Subsystem Master Clock (SMCLK)</i>	444
16.1.8	<i>Auxiliary Clock (ACLK)</i>	444
16.1.9	<i>Default Settings on Power-Up</i>	446
16.1.10	<i>CS Configuration Registers</i>	447
16.2	CONFIGURING THE CS ON THE MSP430FR2355	456
17:	LOW POWER MODES	463
17.1	OVERVIEW OF THE MSP430FR2355's LOW POWER MODES	463
17.1.1	<i>Active Mode (AM)</i>	464
17.1.2	<i>Low Power Mode 0 (LPM0): CPU OFF</i>	464
17.1.3	<i>Low Power Mode 3 (LPM3): Standby</i>	464

17.1.4	<i>Low Power Mode 4 (LPM4): Off</i>	464
17.1.5	<i>Low Power Mode 3.5 (LPM3.5): RTC Only</i>	465
17.1.6	<i>Low Power Mode 4.5 (LPM4.5): Shutdown</i>	465
17.1.7	<i>Example of Putting the MSP430FR2355 into Low Power Mode</i>	465
APPENDIX A: CONCEPT CHECK SOLUTIONS		469
REFERENCES		471
INDEX		473