

# Table of Contents

<b>About the Author</b> .....	<b>xv</b>
<b>About the Technical Reviewer</b> .....	<b>xvii</b>
<b>Acknowledgments</b> .....	<b>xix</b>
<b>Introduction</b> .....	<b>xxi</b>
<b>Chapter 1: Getting Started</b> .....	<b>1</b>
About the ARM Processor .....	2
What You Will Learn .....	3
Why Use Assembly .....	4
Tools You Need .....	7
Computers and Numbers .....	8
ARM Assembly Instructions .....	11
CPU Registers .....	12
ARM Instruction Format .....	13
Raspberry Pi Memory .....	15
About the GCC Assembler .....	16
Hello World .....	17
About the Starting Comment .....	20
Where to Start .....	20
Assembly Instructions .....	21
Data .....	22

## TABLE OF CONTENTS

Calling Linux .....	22
Reverse Engineering Our Program .....	23
Summary.....	26
<b>Chapter 2: Loading and Adding .....</b>	<b>27</b>
Negative Numbers .....	27
About Two's Complement .....	27
About Gnome Programmer's Calculator .....	29
About One's Complement .....	30
Big vs. Little-endian .....	30
About Bi-endian.....	32
Pros of Little-endian .....	32
Shifting and Rotating .....	33
About Carry Flag.....	33
About the Barrel Shifter .....	34
Basics of Shifting and Rotating .....	35
MOV/MVN .....	36
About MOVT .....	36
Register to Register MOV.....	37
The Dreaded Flexible Operand2 .....	37
MVN .....	40
MOV Examples.....	41
ADD/ADC .....	45
Add with Carry.....	47
Summary.....	51

<b>Chapter 3: Tooling Up.....</b>	<b>53</b>
GNU Make .....	53
Rebuilding a File.....	54
A Rule for Building .s files .....	54
Defining Variables.....	55
GDB .....	56
Preparing to Debug.....	56
Beginning GDB.....	58
Source Control and Build Servers .....	63
Git .....	63
Jenkins .....	64
Summary.....	65
<b>Chapter 4: Controlling Program Flow .....</b>	<b>67</b>
Unconditional Branch.....	67
About the CPSR.....	68
Branch on Condition.....	70
About the CMP Instruction .....	71
Loops .....	71
FOR Loops .....	72
While Loops .....	73
If/Then/Else.....	74
Logical Operators.....	75
AND.....	75
EOR.....	76
ORR.....	76
BIC .....	76
Design Patterns.....	77

## TABLE OF CONTENTS

Converting Integers to ASCII .....	78
Using Expressions in Immediate Constants.....	82
Storing a Register to Memory.....	82
Why Not Print in Decimal?.....	83
Performance of Branch Instructions .....	83
More Comparison Instructions.....	84
Summary.....	85
<b>Chapter 5: Thanks for the Memories .....</b>	<b>87</b>
Defining Memory Contents .....	88
Loading a Register .....	92
PC Relative Addressing.....	92
Loading from Memory .....	95
Indexing Through Memory.....	96
Storing a Register .....	107
Double Registers.....	108
Summary.....	108
<b>Chapter 6: Functions and the Stack .....</b>	<b>109</b>
Stacks on Raspbian .....	110
Branch with Link.....	111
Nesting Function Calls .....	112
Function Parameters and Return Values.....	114
Managing the Registers.....	114
Summary of the Function Call Algorithm .....	115
Uppercase Revisited .....	116
Stack Frames.....	121
Stack Frame Example.....	123

Macros ..... 125

    Include Directive ..... 128

    Macro Definition ..... 128

    Labels ..... 129

    Why Macros? ..... 129

    Summary ..... 130

**Chapter 7: Linux Operating System Services ..... 131**

    So Many Services ..... 131

    Calling Convention ..... 132

    Structures ..... 133

    Wrappers ..... 134

    Converting a File to Uppercase ..... 135

        Opening a File ..... 140

        Error Checking ..... 140

        Looping ..... 142

    Summary ..... 143

**Chapter 8: Programming GPIO Pins ..... 145**

    GPIO Overview ..... 145

    In Linux, Everything Is a File ..... 146

    Flashing LEDs ..... 148

    Moving Closer to the Metal ..... 152

    Virtual Memory ..... 153

        About Raspberry Pi 4 RAM ..... 154

    In Devices, Everything Is Memory ..... 154

    Registers in Bits ..... 155

        GPIO Function Select Registers ..... 156

        GPIO Output Set and Clear Registers ..... 158

## TABLE OF CONTENTS

More Flashing LEDs .....	158
Root Access .....	164
Table Driven .....	164
Setting Pin Direction .....	165
Setting and Clearing Pins .....	166
Summary .....	167
<b>Chapter 9: Interacting with C and Python .....</b>	<b>169</b>
Calling C Routines .....	169
Printing Debug Information .....	170
Adding with Carry Revisited .....	173
Calling Assembly Routines from C .....	175
Packaging Our Code .....	178
Static Library .....	178
Shared Library .....	179
Embedding Assembly Code Inside C Code .....	182
Calling Assembly from Python .....	185
Summary .....	187
<b>Chapter 10: Multiply, Divide, and Accumulate .....</b>	<b>189</b>
Multiplication .....	189
Examples .....	191
Division .....	194
Example .....	195
Multiply and Accumulate .....	197
Vectors and Matrices .....	198
Accumulate Instructions .....	199
Example 1 .....	201
Example 2 .....	206
Summary .....	210

<b>Chapter 11: Floating-Point Operations .....</b>	<b>211</b>
About Floating-Point Numbers .....	212
Normalization and NaNs .....	212
Rounding Errors .....	213
Defining Floating-Point Numbers .....	214
FPU Registers .....	214
Function Call Protocol .....	216
About Building .....	217
Loading and Saving FPU Registers .....	217
Basic Arithmetic .....	218
Distance Between Points .....	220
Floating-Point Conversions .....	224
Floating-Point Comparison .....	225
Example .....	227
Summary .....	231
<b>Chapter 12: NEON Coprocessor .....</b>	<b>233</b>
The NEON Registers .....	234
Stay in Your Lane .....	236
Arithmetic Operations .....	237
4D Vector Distance .....	238
3x3 Matrix Multiplication .....	243
Summary .....	248
<b>Chapter 13: Conditional Instructions and Optimizing Code .....</b>	<b>249</b>
Reasons Not to Use Conditional Instructions .....	250
No Conditional Instructions in 64 Bits .....	250
Improved Pipeline .....	250
About Conditional Code .....	251

TABLE OF CONTENTS

- Optimizing the Uppercase Routine.....251
- Simplifying the Range Comparison .....252
- Using a Conditional Instruction.....255
- Restricting the Problem Domain.....256
- Using Parallelism with SIMD .....259
- Summary.....263
  
- Chapter 14: Reading and Understanding Code .....265**
- Raspbian and GCC.....265
- Division Revisited .....267
- Code Created by GCC .....271
- Reverse Engineering and Ghidra.....275
- Summary.....279
  
- Chapter 15: Thumb Code .....281**
- 16-Bit Instruction Format.....282
- Calling Thumb Code .....283
- Thumb-2 Is More than 16 Bits .....285
- IT Blocks .....285
- Uppercase in Thumb-2.....286
- Use the C Compiler .....293
- Summary.....295
  
- Chapter 16: 64 Bits .....297**
- Ubuntu MATE.....297
- About 64 Bits.....298
- More and Bigger Registers .....299
- SP and Zero Register .....300
- Function Call Interface .....301
- Push and Pop Are Gone .....302

Calling Linux Services.....	303
Porting from 32 Bits to 64 Bits.....	303
Porting Uppercase to 64 Bits .....	304
Conditional Instructions .....	308
Example with CSEL.....	309
FPU and the NEON Coprocessors.....	311
Registers .....	311
Instructions.....	312
Comparisons.....	313
Example Using NEON.....	313
Summary.....	315
<b>Appendix A: The ARM Instruction Set.....</b>	<b>317</b>
<b>Appendix B: Linux System Calls .....</b>	<b>327</b>
Linux System Call Numbers .....	327
Linux System Call Error Codes.....	342
<b>Appendix C: Binary Formats .....</b>	<b>347</b>
Integers.....	347
Floating-Point.....	348
Addresses .....	349
64 Bits.....	349
<b>Appendix D: Assembler Directives.....</b>	<b>351</b>
<b>Appendix E: ASCII Character Set .....</b>	<b>353</b>
<b>References.....</b>	<b>365</b>
<b>Index.....</b>	<b>367</b>