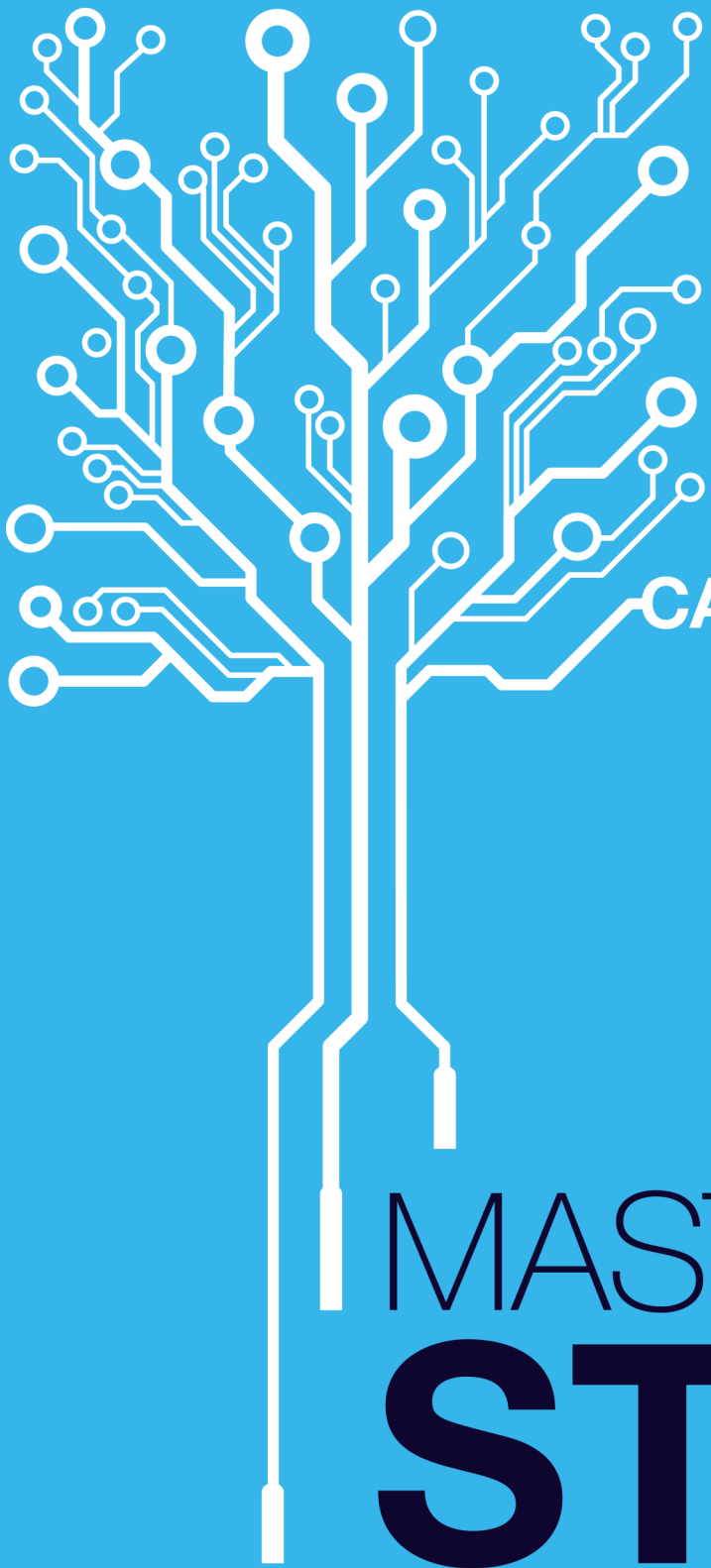
RELEASE **0.25**



CARMINE NOVIELLO

MASTERING STM32

A step-by-step guide to the most complete ARM Cortex-M platform, using a free and powerful development environment based on Eclipse and GCC

Mastering STM32

A step-by-step guide to the most complete ARM Cortex-M platform, using a free and powerful development environment based on Eclipse and GCC

Carmine Noviello

This book is for sale at http://leanpub.com/mastering-stm32

This version was published on 2018-01-03

This is a Leanpub book. Leanpub empowers authors and publishers with the Lean Publishing process. Lean Publishing is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

Tweet This Book!

Please help Carmine Noviello by spreading the word about this book on Twitter!

The suggested hashtag for this book is #MasteringSTM32.

Find out what other people are saying about the book by clicking on this link to search for this hashtag on Twitter:

#MasteringSTM32

Contents

Preface . . . . . **i**
Why Did I Write the Book? i
Who Is This Book For? ii
How to Integrate This Book? iii
How Is the Book Organized? iv
About the Author vii
Errata and Suggestions viii
Book Support viii
How to Help the Author ix
Copyright Disclaimer ix
Credits ix

Acknowledgments . . . . . **x**

I Introduction **1**

1. Introduction to STM32 MCU Portfolio **2**
1.1 Introduction to ARM Based Processors 2
1.1.1 Cortex and Cortex-M Based Processors 4
1.1.1.1 Core Registers 4
1.1.1.2 Memory Map 7
1.1.1.3 Bit-Banding 9
1.1.1.4 Thumb-2 and Memory Alignment 12
1.1.1.5 Pipeline 13
1.1.1.6 Interrupts and Exceptions Handling 15
1.1.1.7 SysTimer 17
1.1.1.8 Power Modes 17
1.1.1.9 CMSIS 19
1.1.1.10 Effective Implementation of Cortex-M Features in the STM32 Portfolio 20
1.2 Introduction to STM32 Microcontrollers 21
1.2.1 Advantages of the STM32 Portfolio.... 22

1.2.2 ....And Its Drawbacks . . . 23
1.3 A Quick Look at the STM32 Subfamilies . . . 24
1.3.1 F0 . . . 26
1.3.2 F1 . . . 27
1.3.3 F2 . . . 28
1.3.4 F3 . . . 30
1.3.5 F4 . . . 32
1.3.6 F7 . . . 33
1.3.7 H7 . . . 35
1.3.8 L0 . . . 36
1.3.9 L1 . . . 37
1.3.10 L4 . . . 38
1.3.11 L4+ . . . 40
1.3.12 W and J STM32 MCUs . . . 41
1.3.13 How to Select the Right MCU for You? . . . 42
1.4 The Nucleo Development Board . . . 45

2. Setting-Up the Tool-Chain . . . 51
2.1 Why Choose Eclipse/GCC as Tool-Chain for STM32 . . . 52
2.1.1 Two Words About Eclipse... . . . 53
2.1.2 ... and GCC . . . 53
2.2 Windows - Installing the Tool-Chain . . . 54
2.2.1 Windows - Eclipse Installation . . . 55
2.2.2 Windows - Eclipse Plug-Ins Installation . . . 57
2.2.3 Windows - GCC ARM Embedded Installation . . . 63
2.2.4 Windows – Build Tools Installation . . . 64
2.2.5 Windows – OpenOCD Installation . . . 64
2.2.6 Windows – ST Tools and Drivers Installation . . . 65
2.2.6.1 Windows – ST-LINK Firmware Upgrade . . . 65
2.3 Linux - Installing the Tool-Chain . . . 66
2.3.1 Linux - Install i386 Run-Time Libraries on a 64-bit Ubuntu . . . 67
2.3.2 Linux - Java Installation . . . 67
2.3.3 Linux - Eclipse Installation . . . 68
2.3.4 Linux - Eclipse Plug-Ins Installation . . . 69
2.3.5 Linux - GCC ARM Embedded Installation . . . 75
2.3.6 Linux - Nucleo Drivers Installation . . . 75
2.3.6.1 Linux – ST-LINK Firmware Upgrade . . . 75
2.3.7 Linux – OpenOCD Installation . . . 76
2.3.8 Linux - ST Tools Installation . . . 78
2.4 Mac - Installing the Tool-Chain . . . 80
2.4.1 Mac - Eclipse Installation . . . 81
2.4.2 Mac - Eclipse Plug-Ins Installation . . . 83
2.4.3 Mac - GCC ARM Embedded Installation . . . 88

2.4.4 Mac - Nucleo Drivers Installation . . . 89
2.4.4.1 Mac – ST-LINK Firmware Upgrade . . . 89
2.4.5 Mac – OpenOCD Installation . . . 90
2.4.6 Mac - ST Tools Installation . . . 92

3. Hello, Nucleo! . . . 95
3.1 Get in Touch With the Eclipse IDE . . . 95
3.2 Create a Project . . . 99
3.3 Connecting the Nucleo to the PC . . . 106
3.4 Flashing the Nucleo using STM32CubeProgrammer . . . 107
3.5 Understanding the Generated Code . . . 108

4. STM32CubeMX Tool . . . 111
4.1 Introduction to CubeMX Tool . . . 111
4.1.1 Pinout View . . . 115
4.1.1.1 Chip View . . . 115
4.1.1.2 IP Tree Pane . . . 117
4.1.2 Clock View . . . 119
4.1.3 Configuration View . . . 120
4.1.4 Power Consumption Calculator View . . . 121
4.2 Project Generation . . . 122
4.2.1 Generate C Project with CubeMX . . . 122
4.2.1.1 Understanding Generated Code . . . 124
4.2.2 Create Eclipse Project . . . 126
4.2.3 Importing Generated Files Into the Eclipse Project Manually . . . 129
4.2.4 Importing Files Generated With CubeMX Into the Eclipse Project Automatically . . . 134
4.3 Understanding Generated Application Code . . . 135
4.3.1 Add Something Useful to the Firmware . . . 140
4.4 Downloading Book Source Code Examples . . . 141

5. Introduction to Debugging . . . 145
5.1 Getting Started With OpenOCD . . . 145
5.1.1 Launching OpenOCD . . . 146
5.1.1.1 Launching OpenOCD on Windows . . . 147
5.1.1.2 Launching OpenOCD on Linux and MacOS X. . . . 148
5.1.2 Connecting to the OpenOCD Telnet Console . . . 150
5.1.3 Configuring Eclipse . . . 151
5.1.4 Debugging in Eclipse . . . 158
5.2 ARM Semihosting . . . 163
5.2.1 Enable Semihosting on a New Project . . . 163
5.2.1.1 Using Semihosting With C Standard Library . . . 166
5.2.2 Enable Semihosting on an Existing Project . . . 169

5.2.3 Semihosting Drawbacks . . . 170
5.2.4 Understanding How Semihosting Works . . . 170

II Diving into the HAL . . . 175

6. GPIO Management . . . 176
6.1 STM32 Peripherals Mapping and HAL *Handlers* . . . 176
6.2 GPIOs Configuration . . . 181
6.2.1 GPIO Mode . . . 183
6.2.2 GPIO Alternate Function . . . 185
6.2.3 Understanding GPIO Speed . . . 186
6.3 Driving a GPIO . . . 190
6.4 De-initialize a GPIO . . . 190

7. Interrupts Management . . . 192
7.1 NVIC Controller . . . 192
7.1.1 Vector Table in STM32 . . . 193
7.2 Enabling Interrupts . . . 197
7.2.1 External Lines and NVIC . . . 197
7.2.2 Enabling Interrupts With CubeMX . . . 201
7.3 Interrupt Lifecycle . . . 203
7.4 Interrupt Priority Levels . . . 207
7.4.1 Cortex-M0/0+ . . . 207
7.4.2 Cortex-M3/4/7 . . . 212
7.4.3 Setting Interrupt Priority in CubeMX . . . 218
7.5 Interrupt Re-Entrancy . . . 219
7.6 Mask All Interrupts at Once or an a Priority Basis . . . 220

8. Universal Asynchronous Serial Communications . . . 224
8.1 Introduction to UARTs and USARTs . . . 224
8.2 UART Initialization . . . 228
8.2.1 UART Configuration Using CubeMX . . . 235
8.3 UART Communication in *Polling Mode* . . . 236
8.3.1 Installing a Serial Console in Windows . . . 240
8.3.2 Installing a Serial Console in Linux and MacOS X . . . 243
8.4 UART Communication in *Interrupt Mode* . . . 244
8.4.1 UART Related Interrupts . . . 245
8.5 Error Management . . . 252
8.6 I/O Retargeting . . . 254

9. DMA Management . . . 258
9.1 Introduction to DMA . . . 258

9.1.1 The Need of a DMA and the Role of the Internal Buses . . . 259
9.1.2 The DMA Controller . . . 262
9.1.2.1 The DMA Implementation in F0/F1/F3/L1 MCUs . . . 263
9.1.2.2 The DMA Implementation in F2/F4/F7 MCUs . . . 267
9.1.2.3 The DMA Implementation in L0/L4 MCUs . . . 270
9.2 HAL_DMA Module . . . 271
9.2.1 DMA_HandleTypeDef in F0/F1/F3/L0/L1/L4 HALs . . . 271
9.2.2 DMA_HandleTypeDef in F2/F4/F7 HALs . . . 274
9.2.3 DMA_HandleTypeDef in L0/L4 HALs . . . 277
9.2.4 How to Perform Transfers in Polling Mode . . . 277
9.2.5 How to Perform Transfers in Interrupt Mode . . . 280
9.2.6 How to Perform *Peripheral-To-Peripheral* Transfers . . . 282
9.2.7 Using the HAL_UART Module With DMA Mode Transfers . . . 283
9.2.8 Miscellaneous Functions From HAL_DMA and HAL_DMA_Ex Modules . . . 286
9.3 Using CubeMX to Configure DMA Requests . . . 287
9.4 Correct Memory Allocation of DMA Buffers . . . 288
9.5 A Case Study: The DMA *Memory-To-Memory* Transfer Performance Analysis . . . 289

10. Clock Tree . . . 294
10.1 Clock Distribution . . . 294
10.1.1 Overview of the STM32 Clock Tree . . . 296
10.1.1.1 The Multispeed Internal RC Oscillator in STM32L Families . . . 300
10.1.2 Configuring Clock Tree Using CubeMX . . . 301
10.1.3 Clock Source Options in Nucleo Boards . . . 303
10.1.3.1 OSC Clock Supply . . . 303
10.1.3.2 OSC 32kHz Clock Supply . . . 304
10.2 Overview of the HAL_RCC Module . . . 305
10.2.1 Compute the Clock Frequency at Run-Time . . . 307
10.2.2 Enabling the *Master Clock Output* . . . 308
10.2.3 Enabling the *Clock Security System* . . . 308
10.3 HSI Calibration . . . 309

11. Timers . . . 311
11.1 Introduction to Timers . . . 311
11.1.1 Timer Categories in an STM32 MCU . . . 312
11.1.2 Effective Availability of Timers in the STM32 Portfolio . . . 314
11.2 Basic Timers . . . 316
11.2.1 Using Timers in *Interrupt Mode* . . . 319
11.2.1.1 Time Base Generation in *Advanced* Timers . . . 322
11.2.2 Using Timers in *Polling Mode* . . . 322
11.2.3 Using Timers in *DMA Mode* . . . 323
11.2.4 Stopping a Timer . . . 325
11.2.5 Using CubeMX to Configure a *Basic Timer* . . . 325

11.3 General Purpose Timers . . . 326
11.3.1 Time Base Generator With External Clock Sources . . . 326
11.3.1.1 External Clock Mode 2 . . . 328
11.3.1.2 External Clock Mode 1 . . . 332
11.3.1.3 Using CubeMX to Configure the Source Clock of a *General Purpose* Timer . . . 337
11.3.2 Master/Slave Synchronization Modes . . . 338
11.3.2.1 Enable Trigger-Related Interrupts . . . 343
11.3.2.2 Using CubeMX to Configure the Master/Slave Synchronization . . . 343
11.3.3 Generate Timer-Related Events by Software . . . 344
11.3.4 Counting Modes . . . 346
11.3.5 Input Capture Mode . . . 347
11.3.5.1 Using CubeMX to Configure the Input Capture Mode . . . 354
11.3.6 Output Compare Mode . . . 355
11.3.6.1 Using CubeMX to Configure the Output Compare Mode . . . 360
11.3.7 Pulse-Width Generation . . . 360
11.3.7.1 Generating a Sinusoidal Wave Using PWM . . . 364
11.3.7.2 Using CubeMX to Configure the PWM Mode . . . 369
11.3.8 One Pulse Mode . . . 370
11.3.8.1 Using CubeMX to Configure the OPM Mode . . . 372
11.3.9 Encoder Mode . . . 373
11.3.9.1 Using CubeMX to Configure the *Encoder Mode* . . . 378
11.3.10 Other Features Available in *General Purpose* and *Advanced* Timers . . . 379
11.3.10.1 *Hall Sensor* Mode . . . 379
11.3.10.2 Combined Three-Phase PWM Mode and Other Motor-Control Related Features . . . 380
11.3.10.3 Break Input and Locking of Timer Registers . . . 380
11.3.10.4 Preloading of Auto-Reload Register . . . 380
11.3.11 Debugging and Timers . . . 381
11.4 SysTick Timer . . . 382
11.4.1 Use Another Timer as System Timebase Source . . . 383
11.5 A Case Study: How to Precisely Measure Microseconds With STM32 MCUs . . . 384

12. Analog-To-Digital Conversion . . . 390
12.1 Introduction to SAR ADC . . . 390
12.2 HAL_ADC Module . . . 395
12.2.1 Conversion Modes . . . 398
12.2.1.1 Single-Channel, Single Conversion Mode . . . 398
12.2.1.2 Scan Single Conversion Mode . . . 398
12.2.1.3 Single-Channel, Continuous Conversion Mode . . . 399
12.2.1.4 Scan Continuous Conversion Mode . . . 399
12.2.1.5 Injected Conversion Mode . . . 400
12.2.1.6 Dual Modes . . . 400

12.2.2 Channel Selection . . . 401
12.2.3 ADC Resolution and Conversion Speed . . . 402
12.2.4 A/D Conversions in Polling Mode . . . 402
12.2.5 A/D Conversions in Interrupt Mode . . . 406
12.2.6 A/D Conversions in DMA Mode . . . 407
12.2.6.1 Convert Multiple Times the Same Channel in DMA Mode . . . 411
12.2.6.2 Multiple and not Continuous Conversions in DMA Mode . . . 411
12.2.6.3 Continuous Conversions in DMA Mode . . . 411
12.2.7 Errors Management . . . 411
12.2.8 Timer-Driven Conversions . . . 412
12.2.9 Conversions Driven by External Events . . . 415
12.2.10 ADC Calibration . . . 416
12.3 Using CubeMX to Configure ADC Peripheral . . . 416

13. Digital-To-Analog Conversion . . . 419
13.1 Introduction to the DAC Peripheral . . . 419
13.2 HAL_DAC Module . . . 422
13.2.1 Driving the DAC Manually . . . 423
13.2.2 Driving the DAC in DMA Mode Using a Timer . . . 425
13.2.3 Triangular Wave Generation . . . 429
13.2.4 Noise Wave Generation . . . 430

14. I²C . . . 432
14.1 Introduction to the I²C specification . . . 432
14.1.1 The I²C Protocol . . . 434
14.1.1.1 START and STOP Condition . . . 435
14.1.1.2 Byte Format . . . 435
14.1.1.3 Address Frame . . . 435
14.1.1.4 Acknowledge (ACK) and Not Acknowledge (NACK) . . . 436
14.1.1.5 Data Frames . . . 437
14.1.1.6 Combined Transactions . . . 437
14.1.1.7 Clock Stretching . . . 438
14.1.2 Availability of I²C Peripherals in STM32 MCUs . . . 439
14.2 HAL_I2C Module . . . 440
14.2.1 Using the I²C Peripheral in *Master Mode* . . . 443
14.2.1.1 I/O MEM Operations . . . 451
14.2.1.2 Combined Transactions . . . 453
14.2.1.3 A Note About the Clock Configuration in STM32F0/L0/L4 families 455
14.2.2 Using the I²C Peripheral in *Slave Mode* . . . 455
14.3 Using CubeMX to Configure the I²C Peripheral . . . 461

15. SPI . . . 463
15.1 Introduction to the SPI Specification . . . 463

15.1.1 Clock Polarity and Phase . . . 466
15.1.2 Slave Select Signal Management . . . 467
15.1.3 SPI *TI Mode* . . . 468
15.1.4 Availability of SPI Peripherals in STM32 MCUs . . . 469
15.2 HAL_SPI Module . . . 470
15.2.1 Exchanging Messages Using SPI Peripheral . . . 472
15.2.2 Maximum Transmission Frequency Reachable using the CubeHAL . . . 474
15.3 Using CubeMX to Configure SPI Peripheral . . . 474

16. Cyclic Redundancy Check . . . 475
16.1 Introduction to CRC Computing . . . 475
16.1.1 CRC Calculation in STM32F1/F2/F4/L1 MCUs . . . 478
16.1.2 CRC Peripheral in STM32F0/F3/F7/L0/L4 MCUs . . . 480
16.2 HAL_CRC Module . . . 481

17. IWDG and WWDG Timers . . . 485
17.1 The *Independent Watchdog* Timer . . . 485
17.1.1 Using the CubeHAL to Program IWDG Timer . . . 486
17.2 The *System Window Watchdog* Timer . . . 487
17.2.1 Using the CubeHAL to Program WWDG Timer . . . 489
17.3 Detecting a System Reset Caused by a Watchdog Timer . . . 491
17.4 Freezing Watchdog Timers During a Debug Session . . . 491
17.5 Selecting the Right Watchdog Timer for Your Application . . . 491

18. Real-Time Clock . . . 493
18.1 Introduction to the RTC Peripheral . . . 493
18.2 HAL_RTC Module . . . 495
18.2.1 Setting and Retrieving the Current Date/Time . . . 496
18.2.1.1 Correct Way to Read Date/Time Values . . . 498
18.2.2 Configuring Alarms . . . 500
18.2.3 Periodic Wakeup Unit . . . 502
18.2.4 Timestamp Generation and Tamper Detection . . . 503
18.2.5 RTC Calibration . . . 504
18.2.5.1 RTC Coarse Calibration . . . 504
18.2.5.2 RTC Smooth Calibration . . . 505
18.2.5.3 Reference Clock Detection . . . 506
18.3 Using the Backup SRAM . . . 507

III Advanced topics . . . 508

19. Power Management . . . 509
19.1 Power Management in Cortex-M Based MCUs . . . 509

19.2 How Cortex-M MCUs Handle *Run* and *Sleep* Modes . . . 510
19.2.1 Entering/exiting sleep modes . . . 513
19.2.1.1 Sleep-On-Exit . . . 515
19.2.2 *Sleep* Modes in Cortex-M Based MCUs . . . 516
19.3 Power Management in STM32F Microcontrollers . . . 516
19.3.1 Power Sources . . . 517
19.3.2 Power Modes . . . 518
19.3.2.1 Run Mode . . . 518
19.3.2.1.1 Dynamic Voltage Scaling in STM32F4/F7 MCUs . . . 519
19.3.2.1.2 Over/Under-Drive Mode in STM32F4/F7 MCUs . . . 520
19.3.2.2 Sleep Mode . . . 520
19.3.2.3 Stop Mode . . . 521
19.3.2.4 Standby Mode . . . 522
19.3.2.5 Low-Power Modes Example . . . 522
19.3.3 An Important Warning for STM32F1 Microcontrollers . . . 527
19.4 Power Management in STM32L Microcontrollers . . . 528
19.4.1 Power Sources . . . 528
19.4.2 Power Modes . . . 530
19.4.2.1 Run Modes . . . 530
19.4.2.2 Sleep Modes . . . 532
19.4.2.2.1 Batch Acquisition Mode . . . 533
19.4.2.3 Stop Modes . . . 533
19.4.2.4 Standby Modes . . . 534
19.4.2.5 Shutdown Mode . . . 535
19.4.3 Power Modes Transitions . . . 536
19.4.4 Low-Power Peripherals . . . 536
19.4.4.1 LPUART . . . 536
19.4.4.2 LPTIM . . . 537
19.5 Power Supply Supervisors . . . 537
19.6 Debugging in Low-Power Modes . . . 538
19.7 Using the CubeMX Power Consumption Calculator . . . 538
19.8 A Case Study: Using Watchdog Timers With Low-Power Modes . . . 540

20. Memory layout . . . 541
20.1 The STM32 Memory Layout Model . . . 541
20.1.1 Understanding Compilation and Linking Processes . . . 543
20.2 The Really Minimal STM32 Application . . . 546
20.2.1 ELF Binary File Inspection . . . 550
20.2.2 `.data` and `.bss` Sections Initialization . . . 552
20.2.2.1 A Word About the `COMMON` Section . . . 559
20.2.3 `.rodata` Section . . . 560
20.2.4 Stack and Heap Regions . . . 562
20.2.5 Checking the Size of Heap and Stack at Compile-Time . . . 565

20.2.6 Differences With the Tool-Chain Script Files . . . 566
20.3 How to Use the CCM Memory . . . 568
20.3.1 Relocating the *vector table* in CCM Memory . . . 571
20.4 How to Use the MPU in Cortex-M0+/3/4/7 Based STM32 MCUs . . . 574
20.4.1 Programming the MPU With the CubeHAL . . . 578

21. Flash Memory Management . . . 582
21.1 Introduction to STM32 Flash Memory . . . 582
21.2 The HAL_FLASH Module . . . 586
21.2.1 Flash Memory Unlocking . . . 586
21.2.2 Flash Memory Erasing . . . 586
21.2.3 Flash Memory Programming . . . 588
21.2.4 Flash Read Access During Programming and Erasing . . . 589
21.3 Option Bytes . . . 589
21.3.1 Flash Memory Read Protection . . . 591
21.4 Optional OTP and True-EEPROM Memories . . . 593
21.5 Flash Read Latency and the ART™ Accelerator . . . 594
21.5.1 The Role of the TCM Memories in STM32F7 MCUs . . . 597
21.5.1.1 How to Access Flash Memory Through the TCM Interface . . . 603
21.5.1.2 Using CubeMX to Configure Flash Memory Interface . . . 604

22. Booting Process . . . 606
22.1 The Cortex-M Unified Memory Layout and the Booting Process . . . 606
22.1.1 Software *Physical Remap* . . . 607
22.1.2 Vector Table Relocation . . . 608
22.1.3 Running the Firmware From SRAM Using the GNU MCU Eclipse Toolchain 610
22.2 Integrated Bootloader . . . 611
22.2.1 Starting the Bootloader From the On-Board Firmware . . . 613
22.2.2 The Booting Sequence in the GNU MCU Eclipse Tool-chain . . . 614
22.3 Developing a Custom Bootloader . . . 617
22.3.1 *Vector Table* Relocation in STM32F0 Microcontrollers . . . 628
22.3.2 How to Use the flasher.py Tool . . . 631

23. Running FreeRTOS . . . 633
23.1 Understanding the Concepts Underlying an RTOS . . . 634
23.2 Introduction to FreeRTOS and CMSIS-RTOS Wrapper . . . 640
23.2.1 The FreeRTOS Source Tree . . . 641
23.2.1.1 How to Import FreeRTOS Manually . . . 642
23.2.1.2 How to Import FreeRTOS Using CubeMX and CubeMXImporter 643
23.2.1.3 How to Enable FPU Support in Cortex-M4F and Cortex-M7 Cores 645
23.3 Thread Management . . . 645
23.3.1 Thread States . . . 648
23.3.2 Thread Priorities and Scheduling Policies . . . 649

23.3.3 Voluntary Release of the Control . . . 653
23.3.4 The *idle* Thread . . . 653
23.4 Memory Allocation and Management . . . 654
23.4.1 Dynamic Memory Allocation Model . . . 655
23.4.1.1 `heap_1.c` . . . 656
23.4.1.2 `heap_2.c` . . . 657
23.4.1.3 `heap_3.c` . . . 657
23.4.1.4 `heap_4.c` . . . 657
23.4.1.5 `heap_5.c` . . . 658
23.4.1.6 How to Use `malloc()` and Related C Functions With FreeRTOS . . . 658
23.4.1.7 FreeRTOS Heap Definition . . . 659
23.4.2 Static Memory Allocation Model . . . 659
23.4.2.1 *idle* Thread Allocation With Static Memory Allocation Model . . . 660
23.4.3 Memory Pools . . . 660
23.4.4 Stack Overflow Detection . . . 662
23.5 Synchronization Primitives . . . 664
23.5.1 Message Queues . . . 664
23.5.2 Semaphores . . . 668
23.5.3 Thread Signals . . . 671
23.6 Resources Management and Mutual Exclusion . . . 672
23.6.1 Mutexes . . . 672
23.6.1.1 The Priority Inversion Problem . . . 673
23.6.1.2 Recursive Mutexes . . . 674
23.6.2 Critical Sections . . . 675
23.6.3 Interrupt Management With an RTOS . . . 676
23.6.3.1 FreeRTOS API and Interrupt Priorities . . . 677
23.7 Software Timers . . . 678
23.7.1 How FreeRTOS Manages Timers . . . 679
23.8 A Case Study: Low-Power Management With an RTOS . . . 680
23.8.1 The *idle* Thread Hook . . . 680
23.8.2 The Tickless Mode in FreeRTOS . . . 682
23.8.2.1 A Schema for the *tickless* Mode . . . 684
23.8.2.2 A Custom *tickless* Mode Policy . . . 687
23.9 Debugging Features . . . 695
23.9.1 `configASSERT()` Macro . . . 695
23.9.2 Run-Time Statistics and Thread State Information . . . 696
23.10 Alternatives to FreeRTOS . . . 699
23.10.1 ChibiOS . . . 700
23.10.2 Contiki OS . . . 700
23.10.3 OpenRTOS . . . 701

24. Advanced Debugging Techniques . . . 702
24.1 Understanding Cortex-M Fault-Related Exceptions . . . 702

24.1.1 The Cortex-M Exception Entrance Sequence and the ARM Calling Convention . . . 704
24.1.1.1 How the GNU MCU Eclipse Tool-chain Handles Fault-Related Exceptions . . . 709
24.1.1.2 How to Interpret the Content of the LR Register on Exception Entrance . . . 711
24.1.2 Fault Exceptions and Faults Analysis . . . 712
24.1.2.1 *Memory Management* Exception . . . 712
24.1.2.2 *Bus Fault* Exception . . . 713
24.1.2.3 *Usage Fault* Exception . . . 714
24.1.2.4 *Hard Fault* Exception . . . 715
24.1.2.5 Enabling Optional Fault Handlers . . . 716
24.1.2.6 Fault Analysis in Cortex-M0/0+ Based Processors . . . 716
24.2 Eclipse Advanced Debugging Features . . . 717
24.2.1 Expressions . . . 717
24.2.1.1 Memory Monitors . . . 718
24.2.2 Watchpoints . . . 719
24.2.3 Instruction Stepping Mode . . . 720
24.2.4 Keil Packs and Peripheral Registers View . . . 721
24.2.5 Core Registers View . . . 724
24.3 Debugging Aids From the CubeHAL . . . 725
24.4 External Debuggers . . . 725
24.4.1 Using SEGGER J-Link for ST-LINK Debugger . . . 727
24.4.2 Using the ITM Interface and SWV Tracing . . . 731
24.5 STM Studio . . . 732
24.6 Debugging two Nucleo Boards Simultaneously . . . 734

25. FAT Filesystem . . . 737
25.1 Introduction to FatFs Library . . . 737
25.1.1 Using CubeMX to Include FatFs Library in Your Projects . . . 740
25.1.1.1 The *Generic Disk Interface* API . . . 741
25.1.1.2 The Implementation of a Driver to Access SD Cards in SPI Mode 742
25.1.2 Relevant FatFs Structures and Functions . . . 743
25.1.2.1 Mounting a Filesystem . . . 743
25.1.2.2 Opening a File . . . 743
25.1.2.3 Reading From/Writing Into a File . . . 744
25.1.2.4 Creating and Opening a Directory . . . 745
25.1.3 How to Configure the FatFs Library . . . 748

26. Develop IoT Applications . . . 751
26.1 Solutions Offered by STM to Develop IoT Applications . . . 752
26.2 The W5500 Ethernet Controller . . . 754
26.2.1 How to Use the W5500 Shield and the `ioLibrary_Driver` Module . . . 758

26.2.1.1 Configuring the SPI Interface . . . 761
26.2.1.2 Configuring the Socket Buffers and the Network Interface . . . 762
26.2.2 Socket APIs . . . 764
26.2.2.1 Handling Sockets in TCP Mode . . . 766
26.2.2.2 Handling Sockets in UDP Mode . . . 766
26.2.3 I/O Retargeting to a TCP/IP Socket . . . 767
26.2.4 Setting up an HTTP Server . . . 769
26.2.4.1 A Web-Based Oscilloscope . . . 772

27. Getting Started With a New Design . . . 785
27.1 Hardware Design . . . 785
27.1.1 PCB Layer Stack-Up . . . 786
27.1.2 MCU Package . . . 787
27.1.3 Decoupling of Power-Supply Pins . . . 788
27.1.4 Clocks . . . 790
27.1.5 Filtering of RESET Pin . . . 791
27.1.6 Debug Port . . . 791
27.1.7 Boot Mode . . . 793
27.1.8 Pay attention to “pin-to-pin” Compatibility... . . . 794
27.1.9 ...And to Selecting the Right Peripherals . . . 795
27.1.10 The Role of CubeMX During the Board Design Stage . . . 795
27.1.11 Board Layout Strategies . . . 798
27.2 Software Design . . . 799
27.2.1 Generating the binary image for production . . . 799

Appendix . . . 802

A. Miscellaneous HAL functions and STM32 features . . . 803
Force MCU reset from the firmware . . . 803
STM32 96-bit Unique CPU ID . . . 803

B. Troubleshooting guide . . . 805
GNU MCU Eclipse Installation Issues . . . 805
Eclipse related issue . . . 805
Eclipse cannot locate the compiler . . . 806
Eclipse continuously breaks at every instruction during debug session . . . 807
The step-by-step debugging is really slow . . . 807
The firmware works only under a debug session . . . 808
STM32 related issue . . . 808
The microcontroller does not boot correctly . . . 808
It is Not Possibile to Flash or to Debug the MCU . . . 810

C. Nucleo pin-out 812
Nucleo-F446RE 813
Arduino compatible headers 813
Morpho headers 813
Nucleo-F411RE 814
Arduino compatible headers 814
Morpho headers 814
Nucleo-F410RB 815
Arduino compatible headers 815
Morpho headers 815
Nucleo-F401RE 816
Arduino compatible headers 816
Morpho headers 816
Nucleo-F334R8 817
Arduino compatible headers 817
Morpho headers 817
Nucleo-F303RE 818
Arduino compatible headers 818
Morpho headers 818
Nucleo-F302R8 819
Arduino compatible headers 819
Morpho headers 819
Nucleo-F103RB 820
Arduino compatible headers 820
Morpho headers 820
Nucleo-F091RC 821
Arduino compatible headers 821
Morpho headers 821
Nucleo-F072RB 822
Arduino compatible headers 822
Morpho headers 822
Nucleo-F070RB 823
Arduino compatible headers 823
Morpho headers 823
Nucleo-F030R8 824
Arduino compatible headers 824
Morpho headers 824
Nucleo-L476RG 825
Arduino compatible headers 825
Morpho headers 825
Nucleo-L152RE 826
Arduino compatible headers 826

Morpho headers . . . 826
Nucleo-L073R8 . . . 827
Arduino compatible headers . . . 827
Morpho headers . . . 827
Nucleo-L053R8 . . . 828
Arduino compatible headers . . . 828
Morpho headers . . . 828

D. STM32 packages . . . 829
LFBGA . . . 829
LQFP . . . 829
TFBGA . . . 830
TSSOP . . . 830
UFBGA . . . 830
UFQFPN . . . 831
VFQFP . . . 831
WLCSP . . . 831

E. History of this book . . . 833
Release 0.1 - October 2015 . . . 833
Release 0.2 - October 28th, 2015 . . . 833
Release 0.2.1 - October 31th, 2015 . . . 833
Release 0.2.2 - November 1st, 2015 . . . 834
Release 0.3 - November 12th, 2015 . . . 834
Release 0.4 - December 4th, 2015 . . . 834
Release 0.5 - December 19th, 2015 . . . 834
Release 0.6 - January 18th, 2016 . . . 835
Release 0.6.1 - January 20th, 2016 . . . 835
Release 0.6.2 - January 30th, 2016 . . . 835
Release 0.7 - February 8th, 2016 . . . 835
Release 0.8 - February 18th, 2016 . . . 836
Release 0.8.1 - February 23th, 2016 . . . 836
Release 0.9 - March 27th, 2016 . . . 836
Release 0.9.1 - March 28th, 2016 . . . 837
Release 0.10 - April 26th, 2016 . . . 837
Release 0.11 - May 27th, 2016 . . . 837
Release 0.11.1 - June 3rd, 2016 . . . 838
Release 0.11.2 - June 24th, 2016 . . . 838
Release 0.12 - July 4th, 2016 . . . 838
Release 0.13 - July 18th, 2016 . . . 839
Release 0.14 - August 12th, 2016 . . . 839
Release 0.15 - September 13th, 2016 . . . 839
Release 0.16 - October 3th, 2016 . . . 840

Release 0.17 - October 24th, 2016 840
Release 0.18 - November 15th, 2016 840
Release 0.19 - November 29th, 2016 841
Release 0.20 - December 28th, 2016 841
Release 0.21 - January 29th, 2017 841
Release 0.22 - May 2nd, 2017 841
Release 0.23 - July 20th, 2017 842
Release 0.24 - December 11th, 2017 842
Release 0.25 - January 3rd, 2018 842