

Index

Note: Page numbers followed by “*f*” and “*t*” refer to figures and tables, respectively.

A

- Active stack pointer selection (ASPEL), 134–135
- ADC_Init function, 137
- Add and subtract with exchange (ASX), 227
- Advanced architectural features, of Cortex-M processor, 6–9
- Advanced high speed bus (AHB), 12–13, 163–164
 - AHB bus matrix lite, 259–260
 - AHB lite bus interface, 163–164
- Advanced peripheral bus (APB), 163
- Alias registers, 55–56, 55*f*
- ANSI standard, 40–41
- Application interrupt and reset control (AIRC), 68*t*, 93
- Application programming interface (API), 18
- Application program status register (APSR), 55–56
- Architecture of Cortex-M processor, 53
 - bit banding (exercise), 71–72
 - bit manipulation, 69–71
 - bootloader, 94–95
 - exercise, 95–99
 - Cortex-M0+, 106–107
 - Cortex-M0, 105–106
 - Cortex-M4, 104–105
 - dedicated bit manipulation instructions, 72
 - exception model, 86–89
 - NVIC late arriving, 88
 - NVIC POP preemption, 89
 - NVIC tail chaining, 87–88
 - exceptions, 82–84
 - bus fault, 83
 - enabling fault exceptions, 84
 - hard fault, 83
 - memory manager fault, 83
 - usage fault, 82–83
 - groups and subgroup, 85–86
 - IF THEN blocks, 57–59
 - instruction set, 53–54
 - interrupt handling
 - entry, 74–76
 - exit, 76–77
 - interrupts and multicycle instructions, 57
 - memory barrier instructions, 67
 - memory map and busses, 65–67
 - nested vector interrupt controller, 73–74
 - operating modes, 74
 - power management, 99–104
 - low-power modes, 101–104
 - priority and preemption, 84–85
 - program status register (PSR), 55–56
 - programmer’s model and CPU registers, 54–55
 - Q bit and saturated math instructions, 56–57
 - run time priority control, 86
 - saturated math and conditional execution (exercise), 59–64
 - system control block, 67–69
 - systick interrupt (exercise), 77–82
 - systick timer, 73
 - working with multiple interrupts (exercise), 89–94
 - write buffer, 67
- ARM (NXP) MBED module, 18*f*
- ARM 32-bit instruction set, 7–8
- ARM7, 2–5, 7
- ARM9, 2–3, 7
- ARM-based microcontrollers, 14, 163, 163*f*
- ARM instruction set, 7, 53, 53*f*
- ARM ISO Libraries and Keil Microlibrary, 41*t*
- Automatic State Preservation Enable (ASPEN) bit, 220–221
- Auxiliary control, 68*t*

B

- BASEPRI register, 54–55, 86*t*
- Bit banding, 69–70
 - exercise, 71–72
- Bit manipulation, 69–71
 - dedicated bit manipulation instructions, 72, 73*t*
- Blinky project, 20–38
- Block processing, 244*f*, 245
 - FIR filter with (exercise), 245–248
- Bootloader, 94–95
 - exercise, 95–99
- Buffering techniques
 - double/circular buffer, 251
 - FIFO message queue, 252–255

Bus fault address, 68*t*
Bus fault exception, 83

C

Calling tree, 136
Change Processor State (CPS)
 instructions, 134–135
Circular buffer, 251
Clear exclusive (CLREX)
 instruction, 148–149
Clock_1s, 161
CMSIS. *See* Cortex
 microcontroller software
 interface standard (CMSIS)
CMSIS RTOS, developing with, 165
 CMSIS RTX and SVC
 exceptions (exercise),
 191–194
 semaphores, 193–194
 code building and starting
 debugger, 179–180
 creating and managing threads
 (exercise), 174–178
 creating threads, 172–174
 first project, 171–172
 first steps with, 167
 idle demon, 185
 exercise, 185–186
 inter-thread communication,
 186–187
 interrupt signal (exercise),
 190–191
 signals, 186–187
 multiple instances, 178
 multiple thread instances
 (exercise), 179
 project, setting up, 166–167
 starting with RTOS, 169–171
 threads, 167–169
 management and priority, 176
 time delay, 180–181
 time management, 180
 exercise, 181–182
 virtual timers, 182
 exercise, 183–184
 waiting for event, 181
Code building and starting
 debugger, 179–180
Command window, 28
Compiler menu, 47

Configurable fault status, 68*t*
Configuration and control
 register, 68*t*
Connect option, 269
Context switch time, 168–169
CONTROL register, 134–135, 134*f*
Cooperative multitasking, 213
Cooperative thread switching,
 176–177
Co processor Access Control
 Register (CPARC), 220
CoreSight, 261
 CMSIS DAP, 297–298
 CMSIS SVD, 292–293
 exercise, 293–297
 Cortex-M0 + MTB, 298–299
 debug architecture, 263*f*,
 264–265
 exercise, 265
 debug features, 301
 debugger hardware, 264
 debug limitations, 271
 embedded trace macrocell
 (ETM), 281–286
 error task, 276–277
 hardware configuration, 266
 instrumentation trace macrocell
 (ITM), 272
 setting up of (exercise),
 272–275
 software testing using, with
 RTOS, 276
 software testing with, 278–281
 JTAG hardware socket,
 263–264
 MTB (exercise), 299–301
 processor fault exceptions
 (exercise), 289–292
 software configuration, 266–271
 software test task, 277–278
 system control block debug
 support, 286–288
 tracking faults, 288–289
CoreSight debug architecture, 5–6
Cortex microcontroller software
 interface standard (CMSIS),
 18, 109
 CMSIS DAP specification,
 113–114, 113*f*, 297–298,
 297*f*, 298*f*

CMSIS DSP library, 238–239
 exercise, 240–244
 functions, 239–240
CMSIS RTOS. *See* CMSIS
 RTOS, developing with
CMSIS RTX and SVC
 exceptions, 191–194
core specification, 111–112
coding rules, 114
configuration values, 121*t*
core, 112, 117*f*
core CPU intrinsic instructions,
 127–128, 128*t*
core debug functions, 129–130,
 130*t*
core header files, 121
core register access, 126–127,
 126*t*
core structure, 117
cortex-based microcontrollers,
 110*f*
defined, 109
device header file, 118–120
documentation, 112*f*
DSP algorithms, 113
foundations of, 114
function groups, 122*t*
instrumentation trace unit (ITM)
 (exercise), 130–131
interrupts and exceptions,
 121–125, 122*t*
intrinsic bit manipulation
 (exercise), 128–129
IO qualifiers, 116*t*
MISRA C, 114–116, 114*f*
RTOS specification, 112
SIMD instructions, 129
specifications, 111–112, 111*f*
startup code, 117–118
system code, 118
system viewer description
 (SVD), 113–114, 292–293
 exercise, 293–297
systick function, 122*t*
and user code comparison
 (exercise), 125–126
variable types, 116*t*
Cortex NVIC, 59
Cortex processor operating modes,
 133–136

Cortex profiles, 1–3
 application, 1*f*
 microcontroller, 1*f*
 real time, 1*f*
 Cortex-A profile, 1–2
 Cortex-M0, 1–2, 9–11, 105–106
 Cortex-M0+, 1–2, 11–14,
 106–107
 microtrace buffer (MTB),
 298–299
 Cortex-M1, 1–2
 Cortex-M3, 1–6, 281
 Cortex-M4, 1–2, 14–15,
 104–105, 281
 DSP instructions, 15–16
 Cortex-R processor, 1–2, 1*f*
 CPU ID, 68*t*
 CPU registers, 54–55, 86

D

Data memory synchronization
 barrier (DMD), 67*t*
 Data synchronization barrier
 (DSB), 67*t*
 Data watchpoint and trace (DWT)
 unit, 5–6
 Dbg_sim.ini file, 31
 DCODE bus, 66
 Debug access port (DAP),
 113–114
 CMSIS DAP, 113–114,
 297–298
 Debug menu, 30, 50
 DEEPSLEEP mode, 100–101
 Digital signal controllers (DSC),
 14, 217*f*
 Digital signal processing (DSP) for
 Cortex-M4, 1–2, 217
 algorithms, optimizing
 (exercise), 231–238
 CMSIS DSP library, 238–239
 exercise, 240–244
 functions, 239–240
 data processing techniques,
 244–245
 double/circular buffer, 251
 DSP algorithms, optimizing
 (exercise), 231–238
 DSP instructions, 15–16,
 225–229

FIFO message queue, 252–255
 FIR filter with block processing
 (exercise), 245–248
 fixed point FFT (exercise),
 249–250
 FPU integration, 218–219
 FPU registers, 219–220
 enabling, 220
 exceptions and, 220–221
 exercise, 221–225
 use of, 221
 hardware floating point unit,
 217–218
 instructions, 225–229
 load, balancing, 255
 load, shouldering, 259–260
 Q numbers, fixed point DSP
 with, 248–249
 real-time processing, designing
 for, 250
 RTX IIR (exercise), 256–259
 SIMD instructions, 225–229
 exercise, 229–231
 Direct memory access (DMA),
 162–164, 259–260
 Double buffer, 251
 Downy, Allen B., 196
 Doxygen, 116

E

Embedded trace macrocell (ETM),
 5–6, 281–286
 Erasable programmable read-only
 memory (EPROM), 261,
 261*f*
 Exception model, 86–89
 NVIC late arriving, 88
 NVIC POP preemption, 89
 NVIC tail chaining, 87–88
 Exclusive access, 147–150, 147*t*
 Execution program status register
 (EPSR), 55–56

F

Fast Fourier transform (FFT), 14,
 250
 fixed point FFT (exercise),
 249–250
 Fault exception configuration
 registers, 287*t*

Fault exceptions, 286–288, 287*t*
 bus fault, 83
 enabling, 84
 hard fault, 83
 memory manager fault, 83
 registers, 287*t*
 usage fault, 82–83
 FAULTMASK register, 54–55,
 86*t*, 134–135
 FIFO message queue, 252–255
 Finite impulse response (FIR)
 filter, 14, 231–238, 231*f*
 with block processing (exercise),
 245–248
 Fixed point FFT (exercise), 249–250
 Floating point context address
 register (FPCAR), 220–221
 Floating point status control
 register (FPSCR), 127,
 219–220
 Floating point unit (FPU), 14*f*,
 217–218
 enabling, 220
 exceptions and, 220–221
 exercise, 221–225
 integration, 218–219
 registers, 219–220
 use of, 221
 Freescale freedom board, 18*f*

G

GNU GCC, 17, 17*t*
 Greenhills, 17*t*

H

Handler mode, 74
 Hard disk drive (HDD) control,
 1–2
 Hard fault exception, 83
 Hard fault status, 68*t*
 Hardware floating point unit
 (hardware FPU), 217–218
 HFNMIENA bit, 153

I

IAR embedded workbench for
 ARM, 17*t*
 ICODE bus, 66, 74–75
 Idle demon, 185
 exercise, 185–186

IF THEN condition blocks, 57–59
 In-circuit emulator, 262, 262*f*, 282
 Independent threads, 186
 Infineon Technologies Relax
 board, 18*f*
 Infinite Impulse Response (IIR),
 256–259
 Instruction condition codes, 58*t*
 Instruction synchronization barrier
 (ISB), 67*t*
 Instruction trace window, 37
 Instrumentation trace (ITM),
 129–131, 272
 setting up of (exercise),
 272–275
 software testing using, with
 RTX RTOS, 276
 software testing with, 278–281
 Instrumentation trace macrocell
 (ITM) unit, 5–6
 Integrated development
 environment (IDE), 17
 Interrupt continuable instruction
 (ICI), 55*f*
 field, 57
 Interrupt control and state, 68*t*
 Interrupt handling
 entry, 74–76
 exit, 76–77
 RTOS, 188–189
 Interrupt program status register
 (IPSr), 55–56
 Interrupt service routine (ISR),
 4–5, 74–75, 123, 144, 256
 Interrupts and multicycle
 instructions, 57
 Inter-thread communication,
 186–187
 interrupt signal (exercise),
 190–191
 signals, 186–187
 exercise, 187–189

J

Joint test action group (JTAG),
 5–6, 262
 debug interface, 262–263
 advantage of, 262–263
 disadvantage of, 262–263
 hardware socket, 263–264

K

Keil microcontroller development
 kit for ARM (MDK-ARM),
 17–18, 17*t*
 lite toolchain, installation, 19
 Keil RTX RTOS, 112, 165,
 203–204, 213

L

Lazy stacking, 220–221
 Lazy State Preservation Enable
 (LSPEN) bit, 220–221
 ledSwitcher() threads, 180
 Linker data types, 25*t*
The Little Book of Semaphores,
 196

Load
 balancing, 255
 shouldering, 259–260
 Logic trace window, 33
 Loop unrolling, 235–236, 237*f*
 Low-power modes, 4–5
 configuring, 101–102
 entering, 101
 exercise, 102–104

M

Mailbox (exercise), 209–216
 configuration, 209–210
 Mail queue, 204–205, 207–209,
 281
 Main stack pointer (MSP), 27,
 68–69
 MBED module, 18*f*, 298*f*
 Memory accelerator, 66–67
 Memory barrier instructions, 67
 Memory manager fault
 address, 68*t*
 Memory manager fault exception,
 83, 288
 Memory map and busses, 65–67
 Memory pool, 206–208
 Memory protection unit (MPU),
 6–7, 83, 150–152, 151*f*,
 152*f*, 153*f*
 configuration, 152–156
 exercise, 156–161
 limitations, 162–163
 subregions, 162

Message queue, 204–205,
 207–208, 255
 exercise, 205–206
 Microtrace buffer (MTB), 13*f*, 298
 exercise, 299–301
 μ vision (MicroVision), 18, 25–26
 books tab, 239*f*
 debugger, 26
 IDE, 95
 project, 166
 MIRA (“Motor Industry Research
 Agency”), 114
 MISRA C specification, 114–116
 Move general register to special
 register (MRS), 54–55
 Move special register to general
 register (MSR), 54–55
 Multiple interrupts, working with
 (exercise), 89–94
 Multiply accumulate (MAC), 15
 Multitasking support, 147
 Mutex, 167, 201–202
 caveats, 203–204
 data exchange, 204
 exercise, 202–203

N

Nested vector interrupt controller
 (NVIC), 4–5, 73–74,
 86–87, 121, 133, 144, 269
 group, 121, 124–125
 late arriving, 88
 POP preemption, 89
 tail chaining, 87–88
 No operation (NOP) instructions,
 57–58
 NULL define, 174

O

Operating modes, 74, 133
 AHB lite bus interface, 163–164
 cortex processor operating
 modes, 133–136
 exclusive access, 147–149, 147*t*
 exercise, 149–150
 interprocessor events, 146–147
 memory protection unit (MPU),
 150–152, 151*f*, 152*f*, 153*f*
 limitations, 162–163
 subregions, 162

memory protection unit (MPU)
 configuration, 152–156
 exercise, 156–161
 Pend_SVC exception, 143–144
 example, 144–146
 stack configuration (exercise),
 136–138
 Supervisor Call (SVC)
 instruction, 138–139
 exercise, 140–143
 “Options for Target” dialog, 30,
 38
 OsDelay() function, 184
 osKernelStart function, 170
 OsPriorityNormal, 176
 OsThreadGetId(), 172–173

P

Pend_SVC exception, 143–144
 example, 144–146
 “Pins” boxes, 31–32, 35
 POP preemption, 89
 Power management, 99–104
 low-power modes, 101–104
 Preemptive priority-based
 scheduling, 170
 Preemptive scheduling, 212
 PRIMASK, 54–55, 86*t*
 register, 134–135
 Priority and preemption, 84–85
 Priority group and subgroup
 values, 85–86
 Priority inversion, 213–214
 exercise, 214–216
 PRIVDEFENABLE bit, 153
 Privilege default Enable
 (PRIVDEFENA), 153*f*
 Privileged mode, 6–7, 134, 139
 Process stack pointer (PSP), 27,
 68–69, 134–137
 Program status register (PSR), 27,
 55–56, 229
 application PSR (APSR),
 55–56
 execution PSR (EPSR), 55–56
 GE bit field results, 229
 interrupt PSR (IPSR), 55–56
 Programmer’s model, 54–55
 Project configuration, for
 Cortex-M family, 38–49

Q

Q bit and saturated math
 instructions, 56–57
 Q numbers
 defining, 248
 fixed point DSP with, 248–249

R

Raw Opcode, 27
 Real-time operating systems
 (RTOS), 4–5, 9–10, 18,
 147
 CMSIS RTOS, 112
 on Cortex-M, 165
 interrupt handling, 188–189
 interthread communication,
 186–187
 kernel, 167*f*
 multiple instances, 178
 semaphores, 201
 starting, 169–171
 Real-time processing, designing
 for, 250
 Reduced instruction set computing
 (RISC), 3–4, 53
 Relax board, 18*f*
 Round-robin-based scheduling,
 212–213
 Round-robin preemptive
 scheduling, 213
 RTX, 18, 112
 configuration, 166
 IIR (exercise), 256–259
 RTOS, 252, 276
 Run time priority control, 86

S

Saturated math instructions, 56–57
 and conditional execution,
 59–64
 exercise, 59–64
 Scatter file, 48–49
 Scheduling options, 211
 Semaphores, 193–194
 barrier (exercise), 200–204
 barrier turnstile, 199–200
 caveats, 201
 configuration, 209–210
 cooperative multitasking, 213

mailbox
 exercise, 209–216
 mail queue, 207–209
 memory pool, 206–207
 message queue, 205
 exercise, 205–206
 multiplex, 197
 exercise, 197–198
 mutex, 201–202
 data exchange, 204
 exercise, 202–203
 mutex caveats, 203–204
 preemptive scheduling, 212
 priority inversion, 213–214
 exercise, 214–216
 rendezvous, 198–199
 exercise, 199–200
 round-robin preemptive
 scheduling, 213
 round-robin-based scheduling,
 212–213
 scheduling options, 211
 signaling, 196–197
 exercise, 195–199
 system timer configuration, 211
 thread definition, 210–211
 time configuration, 211
 using semaphores, 196
 SemMultiplex, 198
 __set_BASEPRI() function, 127
 __set_FAULTMASK() function,
 127
 __set_PRIMASK() function, 127
 SigMod task, 259
 Simulation script, 33
 Single instruction multiple data
 (SIMD), 9
 CMSIS intrinsics, 129
 Cortex-M4 DSP and, 225–229
 instructions, 225–229
 exercise, 229–231
 16-bit Thumb instruction
 set, 8, 53
 Size field, 154
 SLEEP mode, 100
 Software development, for Cortex-M
 Family, 17
 blinky project, 20–38
 disassembly window, 27–38
 register window, 27

Software development, for Cortex-M
 Family (*Continued*)
 building first program
 (exercise), 19–20
 hardware debug, 49–52
 installation, 19
 Keil microcontroller
 development kit, 17–18
 project configuration, 38–49
 toolchains, 17, 17*t*
 tutorial exercises, 18–19
Software simulator, 18
Source code window, 28
Special registers, 54–55
SRAM, 6–7, 39
Stack configuration (exercise),
 136–138
STMicroelectronics discovery
 board, 18*f*
Stream processing, 244*f*, 245
Subregion Disable (SRD) field, 162
Subtract and add with exchange
 (SAX), 227
Supervisor Call (SVC), 138–139,
 139*f*
 exercise, 140–143
SVC_DEAD, 142–143
SVC_Handler code, 142
SYSRESET option, 269
System Configuration section, 184
System control block (SCB),
 67–69, 68*t*
 debug support, 286–288

System control register
 (SCR), 101
SystemCoreClockUpdate(), 118
SystemCoreClock variable, 118,
 123
System handler control and state,
 68*t*
System handler priority, 68*t*
SystemInit() function, 118
System timer configuration, 211
System viewer description (SVD)
 file, 113
Systick function, 122*t*
Systick interrupt (exercise),
 77–82, 91
Systick timer, 4–5, 73, 125, 211

T

Tail chaining, of NVIC, 87–88
Tasking VX toolset for ARM, 17*t*
Thread definition, 210–211
Thread mode, 74, 133–134,
 137–139
Threads, 167–169, 200, 213
 creating, 172–174, 177–178,
 214, 277–278
 managing, 174–178
Thread_lock(), 150
Thumb-2, 7, 9–10, 53*f*, 54,
 66–67, 138–139,
 225–226
 DSP instruction set, 225–226
 instruction set, 8

Thumb bit, 59
Thumb instruction set, 53, 53*f*
Time delay, for RTOS, 180–181
 waiting for event, 181
Time management
 exercise, 181–182
Time management, for RTOS, 180
Timeslice configuration, 211
Twatch function, 31–32

U

ULINK2 debugger, 266, 268
Usage fault exception, 82–83

V

VECTKEY field, 124–125
Vector table offset, 68*t*, 95, 99
Virtual timers, 182
 exercise, 183–184

W

Wakeup interrupt controller
 (WIC), 4–5, 100–101,
 100*f*
WEAK labels, 76
WFE instructions, 101–102
WFI instructions, 101–102
Write buffer, 67

X

xPSR. *See* Program status register
 (PSR)