



Newnes

INCLUDES

FREE
NEWNES ONLINE
MEMBERSHIP

EMBEDDED SYSTEMS

WORLD CLASS DESIGNS

- Hand-picked content selected by embedded systems luminary Jack Ganssle
- Real-world best design practices including chapters on FPGAs, DSPs, and microcontrollers
- Covers both hardware and software aspects of embedded systems

Jack Ganssle EDITOR

Contents

<i>About the Editor</i>	<i>ix</i>
<i>About the Authors</i>	<i>xi</i>
<i>Preface</i>	<i>xv</i>
 Chapter 1: Motors	1
1.1 Stepper Motors	2
1.2 DC Motors	21
1.3 Brushless DC Motors	24
1.4 Tradeoffs Between Motors	37
1.5 Motor Torque	40
 Chapter 2: Testing	47
2.1 Why Test?	48
2.2 When to Test?	50
2.3 Which Tests?	52
2.4 When to Stop?	53
2.5 Choosing Test Cases	54
2.6 Testing Embedded Software	57
2.7 Performance Testing	65
2.8 Maintenance and Testing	70
Additional Reading	71
Summary	72
References	72
 Chapter 3: System-Level Design	75
3.1 Dissecting the Requirements Document	77
3.2 Communications	93
3.3 System Priorities	109

3.4	Error Handling	116
3.5	System-Level Design	119
Chapter 4: Some Example Sensor, Actuator, and Control Applications and Circuits (Hard Tasks)		157
4.1	Introduction	158
4.2	E2BUS PC-Host Interface	160
4.3	Host-to-Module Communications Protocol	165
4.4	Speed-Controlled DC Motor with Tach Feedback and Thermal Cutoff	184
4.5	Two-Axis Attitude Sensor Using MEMS Accelerometer	193
4.6	RS-422-Compatible Indicator Panel	202
Chapter 5: Installing and Using a Version Control System		225
5.1	Introduction	226
5.2	The Power and Elegance of Simplicity	227
5.3	Version Control	228
5.4	Typical Symptoms of Not (Fully) Utilizing a Version Control System	228
5.5	Simple Version Control Systems	229
5.6	Advanced Version Control Systems	229
5.7	What Files to Put Under Version Control	230
5.8	Sharing of Files and the Version Control Client	230
5.9	Integrated Development Environment Issues	231
5.10	Graphical User Interface (GUI) Issues	232
5.11	Common Source Code Control Specification	232
5.12	World Wide Web Browser Interface or Java Version Control Client	233
5.13	Bug Tracking	241
5.14	Non-Configuration Management Tools	243
5.15	Closing Comments	244
	Suggested Reading, References, and Resources	245
Chapter 6: Embedded State Machine Implementation		247
6.1	State Machines	248
6.2	An Example	249
6.3	Implementation	251
6.4	Testing	256
6.5	Crank It	256
	References	257
Chapter 7: Firmware Musings		259
7.1	Hacking Peripheral Drivers	260
7.2	Selecting Stack Size	263
7.3	The Curse of Malloc()	265
7.4	Banking	266

7.5	Logical to Physical	267
7.6	Hardware Issues	268
7.7	The Software	269
7.8	Predicting ROM Requirements	271
7.9	RAM Diagnostics.....	271
7.10	Inverting Bits	273
7.11	Noise Issues.....	275
7.12	A Few Notes on Software Prototyping	278
Chapter 8: Hardware Musings		283
8.1	Debuggable Designs.....	284
8.2	Test Points Galore	284
8.3	Resistors.....	286
8.4	Unused Inputs.....	289
8.5	Clocks	291
8.6	Reset.....	293
8.7	Small CPUs	295
8.8	Watchdog Timers.....	299
8.9	Making PCBs	302
8.10	Changing PCBs	304
8.11	Planning	305
Chapter 9: Closed Loop Controls, Rabbits and Hounds		307
9.1	Basic PID Controls	311
9.2	Predictive Controls	315
9.3	Combined Reactive and Predictive Controls	317
9.4	Various PID Enhancements.....	318
9.5	Robot Drive Controls	322
9.6	Tuning Controls	326
9.7	Rabbits Chasing Rabbits	328
9.8	Conclusions	328
Chapter 10: Application Examples		331
10.1	Introduction	332
10.2	Automotive Driver Assistance.....	332
10.3	Baseline JPEG Compression Overview	347
10.4	MPEG-2 Encoding	355
10.5	Code Optimization Study Using Open-Source Algorithms.....	364
Chapter 11: Analog I/Os		375
11.1	Analog Inputs	377
11.2	Reading an ADC	380
11.3	Temperature Measurement Example	386

11.4	Analog Outputs	391
11.5	Temperature Display Example	392
11.6	Analog I/O Module	395
11.7	Internals	396
11.8	Interface Functions	401
11.9	Analog I/O Module, Configuration	417
11.10	How to Use the Analog I/O Module	419
11.11	How to Use the Analog I/O Module, AI #0	421
	References	427
Chapter 12: Optimizing DSP Software		451
12.1	Introduction	452
12.2	What Is Optimization?	452
12.3	The Process	453
12.4	Make the Common Case Fast	456
12.5	Make the Common Case Fast: DSP Architectures	456
12.6	Make the Common Case Fast: DSP Algorithms	459
12.7	Make the Common Case Fast: DSP Compilers	460
12.8	An In-Depth Discussion of DSP Optimization	467
12.9	Direct Memory Access	467
12.10	Managing Internal Memory	477
12.11	Loop Unrolling	478
12.12	Software Pipelining	484
12.13	Programmer Helping Out the Compiler	506
12.14	Profile-Based Compilation	520
12.15	Summary	527
	References	528
Chapter 13: Embedded Processors		529
13.1	Introduction	530
13.2	A Simple Embedded Processor	531
13.3	Soft Core Processors on an FPGA	553
13.4	Summary	554
Index		555