Contents

Foreword **xi**
Acknowledgments **xiii**
Outlines **xv**

1 Introduction **1**
1.1 Introduction to embedded system 1
1.2 Example of embedded system using Athena III PC104 2
1.3 Example of embedded systems using ARDUINO® 3
1.4 Example of embedded system using Raspberry Pi 5
1.5 Example of embedded system using PIC 7
1.6 Motivations 8
1.7 Systematic design approach for prototyping embedded systems 10
References 15

2 Linux®-based embedded system design **17**
2.1 Linux® operating system 17
2.2 Building Linux® for embedded systems 20
2.3 Program layouts in Linux® 25
2.4 System design and architecture 27
2.4.1 Main process design 30
2.4.2 Sensor process design 33
2.4.3 Sensor fusion thread design 37
2.4.4 Control process design 39
2.4.5 Actuator driver design 39
2.4.6 Network communication thread design 40
2.5 Testing of components for control systems 42
2.5.1 Inertial measurement unit 42
2.5.2 DVL sensor unit 47
2.5.3 Image video unit 50
2.5.4 Depth sensor unit 56
2.6 Kalman filter 59
2.7 Graphical user interface 59
References 67

3 Modeling and simulation of embedded underwater vehicle system **69**
3.1 Introduction 69
3.2 Overview of remotely operated underwater vehicle 69

3.3 Dynamics modeling of remotely operated underwater vehicle 71
3.3.1 Hydrodynamic damping model 73
3.3.2 Hydrodynamic-added mass model 79
3.4 Validation of experimental results 84
3.4.1 Heave model identification 84
3.4.2 Yaw model identification 88
3.5 Simulation of remotely operated underwater vehicle model 92
3.6 Simulating external disturbance for remotely operated underwater vehicle model 97
3.7 Launch and recovery process model 101
3.8 Control systems design 102
3.8.1 Sliding-mode control 104
3.8.2 Proposed fuzzy-based genetic algorithm for SMC 105
3.8.3 Proportional-integral-derivative 110
3.9 Remotely operated underwater vehicle sea trial 117
References 119

4 xPC-Target embedded system design 121
4.1 Introduction 121
4.2 Overview of hardware interfacings for simulations testing 123
4.3 Hardware interfacings 124
4.4 Hardware-in-the-loop testing using xPC-Target 134
4.4.1 Create xPC-Target real-time kernel using desktop PC as target PC 136
4.4.2 Create xPC-Target real-time kernel using Athena II-PC104 as target PC 139
4.5 Creating xPC-Target Simulink® block diagrams 142
4.6 Using RS232, analog, and digital I/O in xPC-Target 151
4.7 Infrared sensor model 158
4.8 Incremental encoder model 159
4.9 Identification of a servo DC motor 164
4.10 PID speed control of servo DC motor 168
4.11 Sliding-model speed control of servo DC motor 169
4.12 Linear quadratic regulator 171
4.13 Digital speed control of servo DC motor 175
4.14 Case study: marine robotic vehicle with uncertainties using xPC-Target system 177
4.14.1 System design and architecture 178
4.14.2 Underwater robotic vehicle dynamic model 181
4.14.3 Steady-state thruster's dynamics 183
4.14.4 Underwater robotic vehicle—horizontal subsystem model 188
4.14.5 Controller design 199
4.14.6 Implementation and testing 202
References 207

5 PIC embedded system design 211
5.1 Overview of MPLAB IDE 211
5.2 Intelligent vacuum robot system design 212
5.2.1 System design and architecture 212
5.2.2 Programming and system implementation 216
5.2.3 Testing 234
5.3 Remote temperature-sensing system design for patients 235
5.3.1 System design and architecture 236
5.3.2 Programming and system implementation 240
5.3.3 Testing 241
5.4 Wall-climbing robot system design 243
5.4.1 System design and architecture 245
5.4.2 Programming and system implementation 253
5.4.3 Testing 259
5.5 Magnetic conveyor system design 260
5.5.1 System design and architecture 261
5.5.2 Programming and system implementation 274
5.5.3 Testing 284
References 287

6 ARDUINO® embedded system design 291
6.1 Remotely operated vehicle system design 291
6.1.1 System design and architecture 291
6.1.2 Programming and system implementation 303
6.1.3 Testing 307
6.2 Smart control of marine-tracked vehicle for surveillance 313
6.2.1 System design and architecture 314
6.2.2 Programming and system implementation 320
6.2.3 Testing 325
6.3 A sloth bear-inspired pole-climbing robot 327
6.3.1 System design and architecture 327
6.3.2 Programming and system implementation 335
6.3.3 Testing 343
References 344

7 Raspberry Pi-embedded system design 345
7.1 Fouling detection system 345
7.1.1 System design and architecture 347
7.1.2 Programming and system implementation 367
7.1.3 Testing 381
7.2 Multi-hop microprocessor-based prototype system for remote vibration and image monitoring 387
7.2.1 System design and architecture 388
7.2.2 Programming and system implementation 394
7.2.3 Testing 395

7.3 Face recognition system 399
7.3.1 System design and architecture 399
7.3.2 Programming and system implementation 399
7.3.3 GUI using PyQt 419
7.3.4 Testing 421
References 438

Index 441