Preface

The instructional literature includes many excellent texts on the subject of embedded computing. Appropriately, most such texts, and the courses that use them, are aimed primarily at electrical or computer engineering students. By contrast, this text is designed specifically for mechanical engineering senior undergraduates or first year graduate students. The instruction builds on the typical background of an ME senior, and it focuses on applications of dynamic motion control and digital signal processing that are of particular interest to mechanical engineers.

This course of study is also unique in that learning is organized around a sequence of nine hands-on experiments. Building one upon the other, the experiments proceed from basic interface concepts, through discrete control organization and timing issues, and ultimately to multi-tasked automatic control. Practical information (specific to the lab hardware), as well as, general analytical concepts (applicable to any computing environment) are introduced as necessary to support each experiment.

For students, the text can best be thought of as a journey during which they learn to incorporate (embed) computers into mechanical systems as functional components. They will comprehensively explore issues of user interaction, embedded program organization, timing control, and interface hardware. Along the way, they will learn the process of designing and implementing embedded applications, and become acquainted with the architecture and resources of a modern real-time development environment.

This book evolved during several decades of teaching embedded computing to mechanical engineering students as part of a senior-year curriculum in mechatronics. The goals of the text are first, to provide the embedded computing component for that mechatronics program, culminating in

senior capstone design projects. And second, to provide students with a comprehensive foundation in modern embedded computing that can be used and extended throughout an engineering career.

The vehicle for the lab experiments is the National Instruments myRIO-1900 embedded device, programmed in C from an Eclipse development environment on the student's laptop. This compact system features a two-core Xilinx Z-7010 processor, with an appropriate array of interfaces managed by its associated field programmable gate array, and a real-time linux operating system. Having taught this material (over the years) with at least five other microcomputers, we believe that the myRIO is an ideal platform, both for instruction and as the basis of prototypes in the senior capstone course.

Why C? Of course, the myRIO is also readily programmed in LabVIEW. Why have we chosen C as the programming language for this text? While we are big fans of LabVIEW, and use it in many other courses, in this text we seek to give the student a broad exposure to the concept of an integrated development environment (IDE), useful across a wide array of real-time target microcomputers and applications. An engineer working in this field will eventually become familiar with a variety of languages and development environments. But, C/C++ is the best place for a student to begin, and remains today the lingua franca of embedded computing.

This text is intended for senior mechanical engineering undergraduates. We assume that students will have at least the usual first courses in differential equations, newtonian dynamics, and time/frequency domain analysis of dynamic systems. Background in the subjects of automatic control, instrumentation, and signal processing is useful and will help motivate the material, but is not essential.

Knowledge of computer programming is assumed. However, no specific a priori familiarity with the C programming language is required. By degrees, the student will be introduced to C as he progresses through the lab exercises. We suggest that the student rely on a companion text on C as a reference. Several are listed in the bibliography. The text also recommends that students use MATLAB to evaluate experimental results.

An additional point: Although the book is presented for the classroom, there is no reason why a determined student couldn't master the text and its experiments through individual self-study.

Some ancillary hardware is necessary for the experiments: a keypad and LCD display, an oscilloscope, a brushless DC motor and inertial load with incremental encoder, an appropriate motor amplifier, and an analog signal generator. Although many forms of this equipment can be used, specific

suggestions are detailed in Appendix X.

Before we begin, let's briefly chart this journey into embedded computing by outlining the nine core experiments.

Lab Exercise 00 In this first lab, students set up their development environment and insure that it works properly with the myRIO targets. They exercise debugging techniques, and become familiar with structure of an embedded C program.

Lab Exercises 01, 02, and 03 During this sequence of labs, students learn the elements of C, and build a user interface, including the low level drivers for a keypad and LCD display.

Lab Exercise 04 Students implement a finite state machine, and explore elementary timing elementary embedded computing timing.

Lab Exercise 05 External interrupts are introduced, and their potential advantages and limitations are explored.

Lab Exercise 06 Students develop a general-purpose linear discrete dynamic system as the basis of digital signal processing and control.

Lab Exercise 07 Combining elements of all the previous labs, students implement and analyze a multi-tasked closed-loop velocity control.

Lab Exercise 08 In the final lab, students design, implement, analyze a closed-loop position control. Trajectory planning is also examined.

In each of the final three lab exercises (06, 07, and 08), students compare the performance of the digital implementations with continuous system models.

Finally, the authors wish to acknowledge the many former students, teaching assistants, and colleagues that have influenced this work, including Professors Jens E. Jorgensen, Peter L. Balise, and William R. Murray.