
Programming and Interfacing the 8051 Microcontroller in C and Assembly

Sencer Yeralan, P.E., Ph.D.
Helen Emery

Rigel Press, a Division of Rigel Corporation

PREFACE

This book is the result of a complete rewrite of a previous textbook, *Programming and Interfacing the 8051 Microcontroller*, which is now out of print. Many interesting things have happened since Intel built the first 8051 more than two decades ago, and even more interesting things since we wrote the previous book. First, due to the advancements in personal computers and graphical user interfaces, better software development environments are available for embedded control. Second, microcontrollers have become more powerful and faster as their relative prices have dropped. Perhaps more importantly, larger memory devices are available at lower prices. These factors now make C programming an attractive alternative. Third, with the expansion of the internet, more information is readily available. In particular, manufacturers' data sheets and various hardware and software users' manuals are available for the downloading. This releases a textbook from the task of repeating detailed specifications. Finally, more and more engineers, students, and hobbyists are involved in embedded control. Almost all engineering schools now have a course on embedded control or microcontrollers for non-electrical engineers. The concept of embedded control has become familiar to the public. Moreover, people now expect a product to include a "computer chip." Unlike a few years ago, one no longer needs to define embedded control or a microcontroller. All these changes were anticipated by many in the field. However, the changes were not totally without surprises. Most importantly, we did not anticipate the 8051 to be around much longer after we wrote the previous textbook. New and more powerful microcontroller families were being introduced. We, as well as many in the field, thought that some other microcontroller would essentially make the 8051 architecture obsolete. Although several 8-bit microcontrollers have successfully entered the market, the 8051 family nevertheless continues to enjoy a very large following. In fact, when a microcontroller core needs to be attached to a smart peripheral, such as a USB controller, or an ISA interface, the manufacturers often opt for an 8051. There are several reasons given for the continued use of the 8051. Many people and institutions are familiar with it, or have legacy code still in use. Moreover, there are many good very-low-cost or free support tools in the public. In addition, the 8051 has the most "second sources" of any 8-bit microcontroller. An industrial design with an 8051 reduces concerns of chip availability and cost. Virtually any conceivable peripheral is available next to an 8051 core: extra memory (ROM, RAM, FLASH, EEPROM), ADC, PWM, MAC, I²C, CAN, USB, ARCNET, etc. Newer semiconductor manufacturing techniques have also contributed to the success. It is anticipated that the 8051's will soon reach the 100 MIPS mark. Improving on the original 12 clock cycles per machine cycle design, newer chips take fewer clock cycles per machine cycle. For example, when there are four clock cycles per machine cycle, the processor runs 3 times faster compared to

the original 8051 at the same oscillator frequency. Recently, two companies have announced 8051's achieving one machine cycle per clock cycle.

This book uses Rigel's Integrated Development Environment (IDE) Reads51. Reads51 is free for students and educational institutions. Simply download Reads51 and the examples in this book from www.rigcorp.com. Support material for the book, such as manufacturers' data sheets for the components used in the experiments are also available at this web site. Most of the experiments use the standard 8051. Reads51 version 4.x includes an improved chip simulator with a virtual TTY window and virtual ports. Many experiments may be run in the chip simulator while the user interacts with the ports by clicking on the icons in the virtual I/O window. This is particularly useful for students who develop their code offline. The full impact of the experiments is achieved when actual hardware is used. The book uses Rigel's RMB-S development board with the 80C515 microcontroller, manufactured by Infineon Technologies (formerly Siemens Microelectronics). The 80C515 contains more ports for interfacing, an improved Timer 2 unit (PTRA) for Pulse-Width Modulation (PWM), and an internal Analog-to-Digital Converter (ADC). Circuit diagrams, parts lists, and simple firmware are given in Chapter 7 for those who would like to build their own 8051- or 80515-based board. In summary, you may,

1. Use the chip simulator given with Rigel Corporation's Reads51 Integrated Development Environment (IDE). The latest version of the IDE may be downloaded from www.rigelcorp.com.
2. Build an 8051- or 80515-based microcontroller system as described in this book and run the programs on the system,
3. Use a commercial evaluation system or an in-circuit emulator.

Since the experiments focus on interfacing the microcontroller, a good set of laboratory test and measurement instruments is recommended. A basic voltmeter and a logic probe will go a long way in debugging hardware. A logic probe may be built simply by connecting an LED to a 470-ohm resistor. Laboratory studies, especially for demanding applications, would benefit from a logic analyzer and an oscilloscope. The Hewlett Packard / Agilent Technologies HP54645D mixed signal storage oscilloscope was used in the development of the experiments in the book. Such oscilloscopes are particularly suited for microcontroller circuits, since they display both digital and analog information. The digital and analog signals may be stored and analyzed, similar to features found in logic analyzers. They also measure times and voltages.

The chapters follow a logical order. Both assembly language and C language experiments are presented. Being a lower-level language, assembly is a better way to start working with the microcontroller. With an understanding of assembly language code, writing C code becomes relatively straightforward. Those readers

who wish to write applications in assembly language may skip sections dealing with C. On the other hand, strictly speaking, readers who would like to write the code in C but wish not to deal with assembly may skip sections on assembly. However, these readers are still advised at least to read the sections on assembly language, since a low-level understanding of the microcontroller aids in writing efficient and effective high-level code in the long run. Those who wish to build their own 8051-based microcontroller system are recommended to start from Chapter 1 and read the entire book before they build the system. In our experience, such review provides a useful general understanding, and saves time in the long run. The reader is assumed to be familiar with digital logic (gates and flip/flops).

Chapter 1 gives the fundamental concepts of microcontrollers. It also introduces the properties of the Intel 8051 and Siemens 80C515 microcontrollers.

Chapter 2 discusses Intel's MCS-51 assembly language, which is used by all members of this microcontroller family. The instruction set is partitioned into three logical sets: data transfer instructions, data manipulation instructions, and program flow control instructions. Illustrative code is given for each instruction. The examples focus on a single machine instruction at a time. A brief overview of the ever-expanding family of 8051-based microcontrollers concludes the chapter.

Chapter 3 builds on the basic understanding of assembly language by introducing commonly used techniques. These techniques combine several different types of instructions to accomplish a meaningful task in an effective manner. Many of the examples given are from the monitor programs used by Rigel's 8051 family boards. These techniques include data manipulation and transfer, look-up tables, branching, string manipulation, software timing routines, and arithmetic routines. The examples are not optimized for speed or length, as it is typically done in application notes. Attention here is given to code readability and ease of understanding. A structured programming approach is taken. Tasks that can be identified as independent logical units are implemented as individual subroutines. This makes the programs more hierarchical and easier to dissect and comprehend.

Chapter 4 introduces hardware experiments using assembly language. These experiments may be studied with any evaluation system, provided that a few push buttons, LEDs, seven-segment displays, and a speaker are available. In this book, such interface devices are called User Input/Output Devices (UIOD). A breadboard for prototyping is recommended. Experiments in Chapter 4 deal with the on-chip peripherals. Each experiment focuses on a separate peripheral. Digital input and output operations, counter and timer operations, interrupt routines, and analog-to-digital conversion are presented.

Chapter 5 introduces the C programming language. The hardware experiments given in Chapter 4 are written in C. Comparing C code to assembly code is an instructive effort in and of itself.

Chapter 6 extends the applications to external circuitry. Examples combine many on-chip facilities to accomplish useful tasks. The examples use both C and assembly language. The examples build upon many of the techniques and routines presented in Chapters 3, 4, and 5.

Chapter 7 gives all the information necessary to build an 8051-based microcontroller system. Circuit diagrams and a bill of materials are given. The listing of a skeletal monitor, MINMON is provided along with instructions to extend the monitor. All software listed is available on Rigel's web site www.rigelcorp.com.

Chapter 8 summarizes the steps involved in software development. It suggests further experiments and projects.

Appendix A gives a very brief overview of the Reads51 IDE. Up-to-date information may be found in the various tutorials, quick start sheets and user's guides available on Rigel's web site. Appendix B reviews the C programming language. Appendix B is a good place to start for those who are new to C. The material is not intended to be an exhaustive study of the language. The reader is referred to the numerous textbooks on the subject for in-depth information. Appendix C presents the number systems used in this textbook. Binary, hexadecimal, binary-coded decimal, and two's complement formats are discussed. Appendix D is a compilation of information sources for the 8051. Manufacturers of chips, boards, and test equipment as well as parts suppliers and software companies are listed.

As mentioned, the example code given is not optimized for speed or length. This book does not intend to replace the manufacturer's data books or application notes. It is intended as a comprehensive introductory book. The authors think that the reader may attempt almost any applications development after reading this book. Experienced programmers or persons familiar with microcontrollers in general, but not well acquainted with the 8051 family of microcontrollers may use the book as a cookbook of ideas, example code, and applications circuitry.

The subject of embedded control and microcontrollers is a very rapidly changing field. Even mature architectures like the 8051 display a flurry of activity as new members of the family are introduced and new and more powerful development tools are made available. Much information and many resources are available on the web. We strongly suggest that the reader closely follow the field by periodically checking the relevant web sites. Appendix D lists manufacturers and information sources as of November 2000.

Perhaps the most significant contribution of this textbook is its hands-on approach to the many experiments. Simply implementing the experiments would give an otherwise uninitiated reader a good general feel for embedded control. After many years of developing industrial embedded control systems, we still enjoy constructing experiments for an educational audience. We hope you find it enjoyable and informative. We were surprised to find out how many of our readers actually built their own system from scratch. Hearing from you has always been our greatest satisfaction. Please drop us a line.

Sencer Yeralan and Helen Emery
Gainesville, Florida
October 2000