PIC Microcontroller Projects in C

Basic to Advanced

Dogan Ibrahim

Notice
No responsibility is assumed by the publisher for any injury and/or damage to persons or property as a matter
of products liability, negligence or otherwise, or from any use or operation of any methods, products,
instructions or ideas contained in the material herein. Because of rapid advances in the medical sciences, in
particular, independent verification of diagnoses and drug dosages should be made.

For information on all Newnes publications visit our
website at http://store.elsevier.com/

Working together
to grow libraries in
developing countries

www.elsevier.com • www.bookaid.org

Preface

A microcontroller is a single chip microprocessor system that contains data and program memory, serial and parallel input—output, timers, external and internal interrupts, all integrated into a single chip that can be purchased for as little as $2.00. About 40% of microcontroller applications are in office automation, such as PCs, laser printers, fax machines, intelligent telephones, and so forth. About one-third of microcontrollers are found in consumer electronic goods. Products like CD players, hi-fi equipment, video games, washing machines, cookers, and so on fall into this category. The communications market, automotive market, and the military share the rest of the application areas.

There are many different types of microcontrollers available from many manufacturers. This book is about the PIC18F family of high-end 8-bit microcontrollers, developed and manufactured by Microchip Inc. The highly popular PIC18F45K22 microcontroller is used in the projects in this book. Many simple, intermediate level, and advanced projects are given in the book. Most projects are developed using the highly popular mikroC Pro for PIC compiler as well as the MPLAB XC8 compiler. All the projects are fully documented where the following is given for each project: project description, project hardware (and project block diagram where appropriate), project PDL, project program, and for some projects suggestions are given for possible modifications and improvements. All the projects have been tested and are working.

Knowledge of the C programming language will be useful. Also, familiarity with at least one member of the PIC16F series of microcontrollers will be an advantage. The knowledge of assembly language programming is not required because all the projects in the book are based on using the C language.

This book is written for students, for practicing engineers, and for hobbyists interested in developing microcontroller-based projects using the PIC series of microcontrollers. Attempt has been made to include as many projects as possible, limited only by the size of the book.

Chapter 1 presents the basic features of microcontrollers.

Chapter 2 provides a short tutorial on the C language and then examines the features of the highly popular mikroC Pro for PIC programming language and compiler used in projects in this book.

Chapter 3 is about the MPLB X IDE and the XC8 programming language and compiler. Both the mikroC Pro and the XC8 program listings are given for most projects in the book. The reader should be able to convert easily from one language to the other.

Chapter 4 describes the commonly used program development tools, such as the PDL and flowcharts. Examples are given for both tools.

Chapter 5 gives simple projects using the PIC18F45K22 microcontroller. In this chapter, the projects range from simple LEDs, 7-segment LED displays, LCD displays, sound projects, and so on.

Chapter 6 provides intermediate level projects. The projects in this chapter range from using the interrupts, using a keypad, generating waveforms in real time, serial communications, GPS data decoding, various bus systems, and so on.

Chapter 7 provides more advanced projects. Some of the projects covered in this chapter are using the Bluetooth communication, RFid, real-time clock, using graphics LCDs, SD cards, Ethernet-based projects, using the CAN bus, multitasking in microcontroller systems, stepping motors, and DC motors. Although the projects on motors are not advanced, they are given in this chapter for completeness.

Dogan Ibrahim
London, 2014

Acknowledgments

Microcomputer Systems

Chapter Outline

1.1 Introduction

The term microcomputer is used to describe a system that includes a minimum of a microprocessor, program memory, data memory, and input–output (I/O) module. Some microcomputer systems include additional components such as timers, counters, interrupt processing modules, analog-to-digital converters, serial communication modules, USB modules, and so on. Thus, a microcomputer system can be anything from a large system having hard disks, keyboard, monitor, floppy disks, and printers to a single chip embedded controller.

In this book, we are going to consider only the type of microcomputers that consists of a single silicon chip. Such microcomputer systems are also called microcontrollers and they are used in many everyday household goods such as personal computers, digital watches, microwave ovens, digital TV sets, TV remote control units (CUs), cookers, hi-fi equipment, CD players, personal computers, fridges, etc.

There are a large number of different types of microcontrollers available in the market, developed and manufactured by many companies. In this book, we shall be looking at the programming and system design using the highly popular 8-bit programmable interface controller (PIC) series of microcontrollers manufactured by Microchip Technology Inc (www.microchip.com).

1.2 Microcontroller Systems

A microcontroller is a single chip computer. *Micro* suggests that the device is small, and *controller* suggests that the device can be used in control applications. Another term used for microcontrollers is *embedded controller*, since most of the microcontrollers in industrial, commercial, and domestic applications are built into (or embedded in) the devices they control.

A microprocessor differs from a microcontroller in many ways. The main difference is that a microprocessor requires several other external components for its operation as a computer, such as program memory and data memory, I/O module, and external clock module. A microcontroller on the other hand has all these support chips incorporated inside the same chip. In addition, because of the multiple chip concept, microprocessor-based systems consume considerably more power than the microcontroller-based systems. Another advantage of microcontroller-based systems is that their overall cost is much less than microprocessor-based systems.

All microcontrollers (and microprocessors) operate on a set of instructions (or the user program) stored in their program memories. A microcontroller fetches these instructions from its program memory one by one, decodes these instructions, and then carries out the required operations.

Microcontrollers have traditionally been programmed using the assembly language of the target device. Although the assembly language is fast, it has several disadvantages. An assembly program consists of mnemonics and in general it is difficult to learn and maintain a program written using the assembly language. Also, microcontrollers manufactured by different firms have different assembly languages and the user is required to learn a new language every time a new microcontroller is to be used.

Microcontrollers can also be programmed using high-level languages, such as BASIC, PASCAL, and C. High-level languages have the advantage that it is much easier to learn a high-level language than an assembler language. Also, very large and complex programs can easily be developed using a high-level language. In this book, we shall be learning the programming of high-end 8-bit PIC microcontrollers using two popular C programming languages: the mikroC Pro for PIC, developed by mikroElektronika (www.mikroe.com), and the MPLAB X IDE, developed by Microchip (www.microchip.com).

In general, a single chip is all that is required to have a running microcontroller-based computer system. In practical applications, additional components may be required to allow a microcomputer to interface to its environment. With the advent of the PIC family of microcontrollers, the development time of an electronic project has reduced to several months, weeks, or even hours.

Basically, a microcontroller (or a microprocessor) executes a user program that is loaded in its program memory. Under the control of this program, data are received from external devices (inputs), manipulated, and then sent to external devices (outputs).

For example, in a microcontroller-based fluid level control system, the aim is to control the level of the fluid at a given point. Here, the fluid level is read by the microcomputer via a level sensor device. The program running inside the microcontroller then actuates the pump and the valve and attempts to control the fluid level at the required value. If the fluid

level is low, the microcomputer operates the pump to draw more fluid from the reservoir. In practice, the pump is controlled continuously in order to keep the fluid at the required level. Figure 1.1 shows the block diagram of our simple fluid level control system.

The system shown in Figure 1.1 is a very simplified fluid level control system. In a more sophisticated system we may have a keypad to set the required fluid level, and an LCD to display the current fluid level in the tank. Figure 1.2 shows the block diagram of this more sophisticated fluid level control system.

We can make our design even more sophisticated (see Figure 1.3) by adding an audible alarm to inform us if the fluid level is outside the required point. Also, the actual level at any time can be sent to a PC every second for archiving and further processing. For example, a graph of the daily fluid level changes can be plotted on the PC. Wireless interface (e.g. Bluetooth or RF) or internet connectivity can be added to the system so that the fluid level can be monitored or controlled remotely. Figure 1.4 shows the block diagram with a Bluetooth module attached to the microcontroller.

As you can see, because the microcontrollers are programmable, it is very easy to make the final system as simple or as complicated as we like.

Another example of a microcontroller-based system is the speed control of a direct current (DC) motor. Figure 1.5 shows the block diagram of such a system. Here, a speed sensor device reads current speed of the motor and this is compared with the desired speed
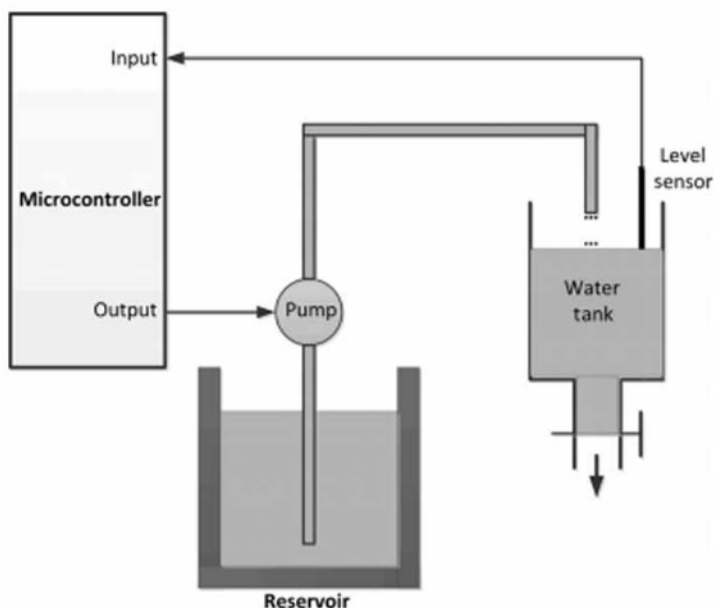

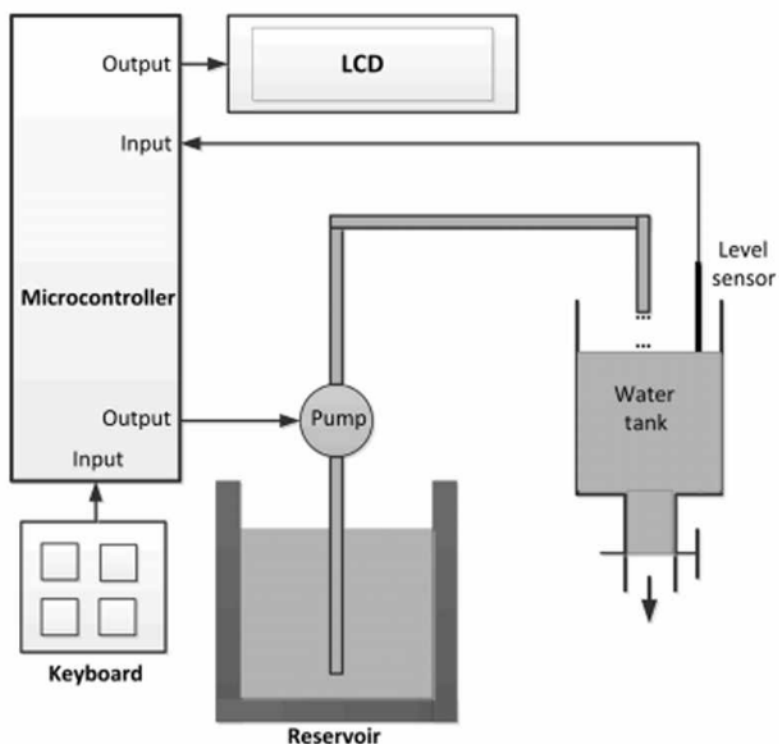**Figure 1.1: Microcontroller-Based Fluid Level Control System.**

Figure 1.2: Fluid Level Control System with a Keypad and LCD.

(which is assumed to be analog). The error signal between the desired and the actual speed is converted into digital and fed to a microcontroller. A control algorithm running on the microcontroller generates control signals that are converted into analog form and are fed to a power amplifier. The output of the power amplifier drives the motor to achieve the desired speed.

Depending upon the nature of the signals the block diagram given in Figure 1.5 can take different shapes. For example, if the output of the speed sensor is digital (e.g. optical encoder) and the set speed is also digital, then there is no need to use the A/D converter at the input of the microcontroller. Also, the D/A converter can be eliminated if the power amplifier can be driven by digital signals.

A microcontroller is a very powerful tool that allows a designer to create sophisticated I/O data manipulation under program control. Microcontrollers are classified by the number of bits they process. The 8-bit devices are the most popular ones and are currently used in most low-cost low-speed microcontroller-based applications. The 16- and 32-bit microcontrollers are much more powerful, but usually more expensive and their use may not be justified in many small to medium-size general purpose applications. In this book, we will be using 8-bit PIC18F series of microcontrollers.
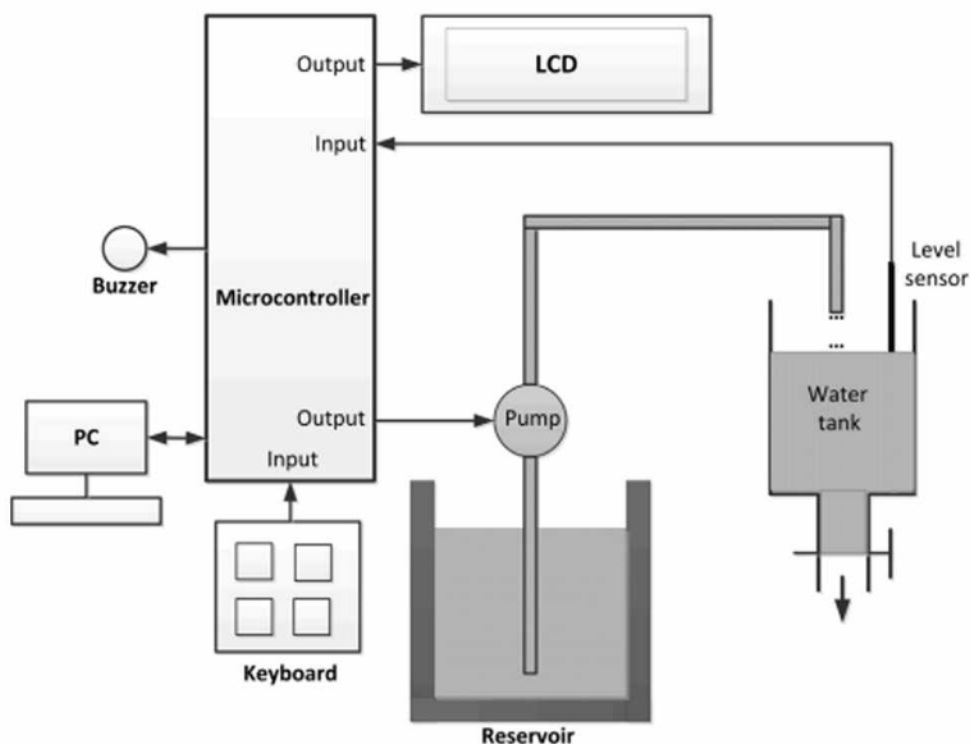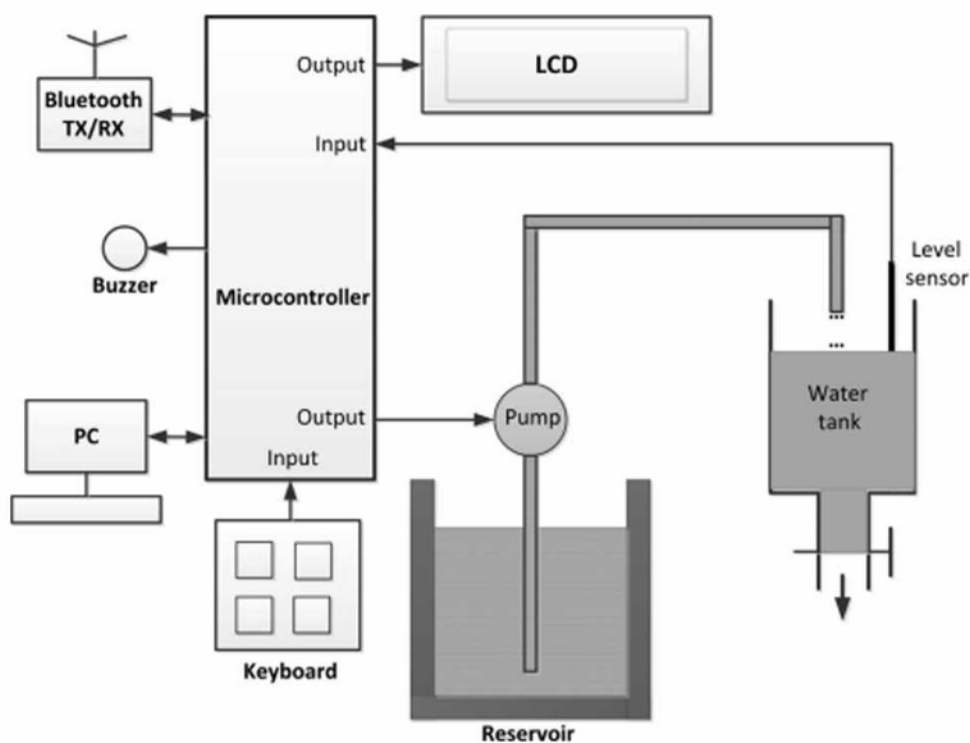
Figure 1.3: More Sophisticated Fluid Level Controller.

Figure 1.4: Using Bluetooth for Remote Monitoring and Control.