

Introduction

This book has been written to empower the reader with knowledge both to interface various hardware devices to ARM microcontrollers and write the software in the C programming language to control these hardware devices.

Although this book targets ARM microcontrollers and a certain ARM7 microcontroller from Atmel, the interfacing circuits and software programming principles apply to many other microcontrollers as well.

Only free or open-source programming tools and software are used throughout the book.

Target Audience

This book has been written for electronic engineers, students and hobbyists and is intended for use:

- 1) As a hardware interfacing and software reference book for microcontrollers in general, but specifically targets ARM microcontrollers.
- 2) For anyone who has learned the C programming language and would like to learn more about interfacing various hardware devices to ARM microcontrollers and how to program these devices.
- 3) As a follow on book to “C Programming for Embedded Microcontrollers” from Elektor that teaches the C programming language on embedded ARM systems. This book now provides more hardware and software examples.

Prerequisites

The reader must be familiar with programs in the C programming language as the software examples are all written in C.

Basic electronics and some knowledge of microcontrollers is required.

Introduction to the ARM Microcontrollers used in this Book

This book uses the **AT91SAM7S256 ARM7** microcontroller from Atmel as its primary microcontroller in interfacing examples. This microcontroller is one from the **AT91SAM7S family** of microcontrollers. All of the original members of this family are pin compatible i.e. the AT91SAM7S512, '256, '128 and '64, varying only in the amount of Flash and SRAM memory on chip. Some newer members were added to this family of microcontrollers that have less memory and/or diminished features, fewer pins and are therefore not pin compatible with the original family.

These microcontrollers are “stand alone” microcontrollers as they do not have an external bus for interfacing to external memory in which programs can be run (although serial memory for data storage can be added to the TWI or SPI serial busses).

AT91SAM7S256 basic features:

- 1) Contains a 32-bit ARM7 microprocessor core and can therefore be programmed using open-source C language programming tools – the GNU ARM toolchain. Commercial tools are also available from several vendors.
- 2) Have 256k Bytes of on-chip Flash memory and 64k Bytes of on-chip SRAM memory. Programs can be run from both of these memories and they can both store data.

- 3) Programs can be loaded onto these microcontrollers via a direct USB connection to a PC, a direct RS-232 connection to a PC or via a JTAG ICE emulator using free software from Atmel running on the PC. No special programmer is required and the microcontroller can be programmed and re-programmed while it is soldered to the circuit board (ISP – In System Programmable).
- 4) Have integrated peripheral devices such as ADC, timers, general purpose I/O controller, SPI port, UARTs, etc. that are multiplexed with other devices that share the same I/O pins. E.g. a pin on the microcontroller can be set up as an input, or an output, or be connected to one of two internal peripheral devices. What device is connected to what pin is configured in the software program. The pins are individually configurable.
- 5) Operate at 3.3V I/O voltage. The core of the microcontroller operates at 1.8V, but a regulator on the chip derives this voltage from the 3.3V supply.
- 6) Have a JTAG port for single step debugging and loading programs using an external hardware device – the JTAG ICE.
- 7) Start running without an external oscillator. When power is applied to the microcontroller, it uses an internal RC oscillator to start running the program. The program must enable the main oscillator that runs from an external crystal.
- 8) Can run up to a speed of 55MHz, but are usually operated at 48MHz using an external 18.432MHz crystal. The internal clock generation circuitry derives the 48 MHz from the 18.432MHz input clock. Running the microcontroller at 48MHz generates the correct clock speed for operating the USB port.

Hardware Requirements

An evaluation board containing a member of the AT91SAM7S family of microcontrollers and preferably the AT91SAM7S256 microcontroller from Atmel is recommended for use with this book, for example the AT91SAM7S-EK evaluation kit from Atmel as described in chapter 1.

Many of the circuits in this book can be built by connecting wires from an evaluation board to a breadboard and assembling the circuit on the breadboard.

In addition to an evaluation board, a JTAG ICE is recommended for debugging. A serial port on the PC or a USB to serial port converter is also required.

Summary of hardware requirements:

- 1) AT91SAM7S256 evaluation board.
- 2) A USB to RS-232 port adapter if a RS-232 port is not available on the PC.
- 3) A JTAG ICE (In-Circuit Emulator) is highly recommended but not essential – e.g. the Atmel SAM-ICE from www.atmel.com or Amontec JTAG key from www.amontec.com.
- 4) An electronic breadboard and/or vero-board on which to build the circuits.

Additional electronic components will be required as described in each chapter.

Software Requirements

The YAGARTO GNU ARM toolchain from www.yagarto.de is used for writing and compiling the C programs in this book. This toolchain runs on Windows XP and Vista. It can be downloaded for free. Alternately the Sourcery G++ Lite GNU ARM toolchain from www.codesourcery.com can be used. This toolchain from CodeSourcery is also available for free.

The software example programs and template files used in this book can be downloaded from the Elektor website at www.elektor.com.

Chapter 1 contains more information on software and hardware setup and debugging.

Advice on Learning a New Microcontroller

The most important source of information when learning a new microcontroller is the microcontroller datasheet. Information can be obtained from the datasheet that will tell you how much memory the microcontroller has, how it boots up, what voltage it operates at, what integrated peripherals it contains and more.

Typical steps to follow when learning a new microcontroller are:

- 1) **Read the datasheet** – Datasheets are long and contain a lot of information that you might not need to read right away. For example you probably won't need every integrated peripheral device that is on the microcontroller, so skip those chapters.
- 2) **Evaluate the software tools** – Determine what software toolchains are available. A microcontroller is of no use unless you can program it. For the hobbyist and student, free software tools are usually a necessity so the availability of GNU tools for the microcontroller are essential.
- 3) **Loading programs to the microcontroller** – Determine how a program is loaded into the microcontroller. In-system programmable (ISP) microcontrollers are a good choice to cut costs, especially if the microcontroller can be programmed through a serial or USB port without any additional hardware required.

- 4) **Hardware tools** – Have a look at what hardware tools such as programmers or in-circuit emulators are available and if they fit your budget.
- 5) **Evaluation board** – Buy an evaluation board so that you can start writing programs for the new microcontroller. Not only must the microcontroller be suitable for your project, but the software toolchain and emulator must be of suitable quality. The microcontroller and programming tools can only be properly evaluated by writing programs and running them.
- 6) **Build Project** – If the microcontroller and tools are suitable, either use the evaluation board, a pre-built board containing the chosen microcontroller or design and build your own microcontroller board for use with your project.

It may be tempting to build your own microcontroller board without buying an evaluation board. This is possible, but the safest way is to buy an evaluation board. The evaluation board is known to be working, so any problems that you have when setting up and using the software toolchain can be more easily determined. If you built your own hardware before you have the toolchain running and are not familiar with the tools, it will be difficult to determine the cause of errors – is it the board that is not working, or did I not set up the program properly?

Approach Taken in this Book

Tested circuits and software are presented in this book to help you to learn microcontroller interfacing. Although not every hardware device that can be interfaced to a microcontroller has been presented, it is hoped that the selection is varied enough to be useful and to teach the principles of interfacing so that you will be able to interface other devices on your own. For this reason, specialised components have been avoided wherever possible.

Hardware and software

Chapter 1 is concerned with getting software tools installed on your computer, getting a hardware platform to work from and compiling and loading C programs to the hardware. It also takes a look at a basic ARM microcontroller circuit and examines the microcontroller selected for use with this book.

Start interfacing and programming

The chapters that follow present interfacing various hardware devices to microcontrollers and software programs that operate the interfaced hardware.

The principles of operation of the hardware devices are included so that you will understand what the software must do. This enables you to convert the software or write your own software to operate the hardware interfaced to any microcontroller.

Operation of the software is also explained, but where it is lacking in detail, you are expected to refer to the example programs that can be downloaded from the support page for this book on the Elektor website. Where reference is made to various microcontroller register and bit fields within these registers, you will need to refer to the microcontroller datasheet as it was considered unnecessary to reproduce these registers in this book.

More advanced microcontrollers

The microcontroller used throughout this book is a fast and powerful ARM microcontroller and a great step up from 8-bit microcontrollers. But what do you do if you need more processing power, I/O pins or more memory? The final chapter in this book presents additional ARM microcontrollers that will solve these problems including some of the latest offerings from Atmel, a major manufacture of such devices (see www.atmel.com).

Download required

Example programs and datasheets that this book refers to must be downloaded from the Elektor website. Inside the zipped file that you download, you will find a webpage (index.html) that contains links to all of the websites and datasheets referenced in the text for easy access to these resources.

Good luck and enjoy interfacing with ARM microcontrollers!