Preface

Last year, 2006 was a very special year, where we witnessed the juxtaposition of four events. It was the 25th anniversary of the IBM PC, which undoubtedly changed the societal perception of computing. Intel announced that it was discontinuing several processor and microcontroller lines. These included the 8051, and 80251, as well as processors up to the Intel486 and the i960 RISC family. Intel had been producing the 8051 for almost three decades – a remarkable record indeed. Last year was reported that for the first time, more 32-bit processors were manufactured than 8-bit processors. And finally, it is widely thought that open-source software had reached wide-spread acceptance.

Two decades ago, I chose the 8051 architecture for classroom instruction. Although I continued to use the 8051 in industrial projects and product design, I found that the solutions offered by the ARM family were becoming extremely cost effective. The point made in the previous paragraph that there now are more 32-bit processors made than 8-bit processors, was visible to all design engineers of embedded systems. Accordingly, in a 2006 workshop in METU on product design, I used the ARM processor and open-source tools, GNU and Eclipse. The same combination was used in my graduate embedded control class at the University of Florida.

From the viewpoint of the instructor, the class requires not only retooling, but a significant paradigm shift. Using the ARM and GNU tools introduces a level of pedagogical complexity not seen when using eight-bit devices. The architecture and instruction set of eight-bit processors can be covered in a single graduate class with relative ease. The related documentation of, say the 8051 is around 100 pages. The MCS-51 architecture has only 255 instructions. In contrast, the ARMv7 instruction set requires about an order of magnitude more material. In fact, with the Thumb instructions, the task of the instructor is formidable. The same goes for the ARM hardware and peripherals. The data sheet for the AT91SAM7S family of microcontrollers that is used in this textbook is about 600 pages. One must add to that several application notes and supplementary material. The observation that there is an order of magnitude difference between the 8-bit to 32-bit devices seems accurate.

Add to this the complexity of the GNU tools. The assembler, linker, C compiler, and make utility documentation spans thousands of pages. Again, add to this not only the documentation of the Eclipse IDE, but also the fact that new and more sophisticated versions of the IDE are continually being introduced. Realistically, an estimate for the material that needs to be taught for a full understanding of all aspects of the task "program the ARM with GNU tools and Eclipse" would be in the order of 10,000 pages. Clearly, no single course or textbook could cover this much material. This is the pedagogical predicament that faces the instructor. It also faces the design engineer who wants to keeps up with the technology. The technology that gives us highly sophisticated 32-bit processors for only a few dollars cannot be readily dismissed or ignored. As I once explained in class, "a few years ago, a few bucks got you from 0 to

FF; now it gets you from 0 to all the way to FFFFFFFF."

This predicament was the main cause for the delay in the compilation of this book. However, the interest for the material became evident when more than 50 students registered for the Fall 2007 offering of the course at the University of Florida. This interest propelled the search for an approach for the instruction of the material.

This book does not pretend to be a replacement for the thousands of pages of documentation. Nor is it a reference book. It does, however, recognize that embedded control has evolved, and that there has been a shift in what is required from the systems design engineer. Rather than "knowing" everything there is about embedded control, the systems engineer must "have the capability to find out" about everything. This is especially the case when components are continually evolving, presenting a very fluid landscape of knowledge and skills. The key concepts must be conveyed. However, rather than following the customary systematic approach that justifies each step as it builds the subject matter from the initial principles, here we resort to a guided exposure. In this sense, the approach is more like case studies popular in business administration and management science schools. We provide an assisted tour of the topics. We not only give examples, but also point out where the information is to be obtained.

As much as possible, we give sources that refer to web sites. The web is a very organic and dynamical medium. Links may and will change. However, with powerful search engines, the references should be relatively easy to track. We also provide a dedicated web page for this book, which will periodically, among other things, list changes to the references. The references necessarily include data books. The appendix contains a section that discusses how data books may be read to achieve a certain level of effectiveness.

We depart from a few traditional defaults, such as using the .s, or .o extension in favor of .asm and .obj. The former are short and efficient for the command-line-oriented experience Unix programmer, but the latter is a better choice for instruction. Similarly, examples are written for ease of understanding rather than code efficiency. Our purpose is to convey the general basic ideas. The current book is far too elementary to convey the advanced ideas of efficient and robust embedded code development.

As in any hands-on approach, one necessarily comes to a point where generalities stop and a certain hardware platform is adopted. In this text, we use the AT91SAM7S series microcontrollers manufactured by Atmel Corporation, and the embedded control board R-A7SJ from Rigel Corporation. With a little work, the experiments could be transposed to work with almost any ARM7 microcontroller or evaluation board.

Sencer Yeralan
September 22, 2007
Gainesville, Florida