June 2002

BOOK REVIEWS

"Embedded Linux Hardware, Software, and Interfacing"
By Craig Hollabaugh, PhD

ISBN: 0672322269

Two reviews by ELC members Joel Williams and Dr. Ian McLoughlin are presented. If you wish to communicate with either the author or the reviewers, please use these e-mail addresses:

Dr. Craig Hollabaugh: craig@hollabaugh.com
Joel Williams: joel@emLinux.com
Dr. Ian McLoughlin: ian.mcloughlin@tait.co.nz

By Joel Williams

Developing Embedded Linux systems is becoming easier all the time. My first experience, back in 1996 took four months to get Linux up and running on a non-PC x86 embedded system. Today, 4 days can be seem long.

Craig Hollabaugh's book "Embedded Linux" shows you just how easy it can be. It takes you through the process of building an embedded Linux system. It covers the development basics of tools, booting, and debugging of a system. It then goes on to show how to solve common I/O interfacing and finishes up with an interesting mix of integrating applications including as web servers and MYSQL. All of this is done through the metaphor of diary of the development project. It is replete with sample code (available for download). For the less 'C' fluent, this can get tedious, but is great for true code jockeys. In some areas, examples are given for a few different CPU architectures. While this is helpful, it gets a bit redundant. You can skim some of this.

The book gives detailed instructions on where to get various downloads and how to install them. This can be particularly helpful, however, the links will probably be stale by the time you get there, and newer versions are sure to have different installation instructions.

If you are a Linux developer newbie, this is a great place to start. There are step-by-step cookbooks that provide a good insight to how to use tools and software. The detailed examples may not be applicable to your development, but the bag of tricks Hollabaugh uses and his approaches to dealing with common development issues are extremely valuable for the new developer.

This book may be all you need to get started developing your embedded system. Then again, maybe not. Let me explain. You can divide embedded systems into two general categories: soft and hard as explained below.

The systems addressed by the Hollabaugh's book is based on using off the shelf CPU cards, for which Linux has already been ported and supported by some sort of board support package or pre-configured software development kit. We can call this category of system "soft embedded". The hardware has ample CPU horsepower, lots of memory, and usually fits into a back plane system such as CompactPCI. Mostly, you do configuration of the support package, rather than hard-core coding.
The tools are also "pre-packaged", a true blessing. Once Linux boots, development essentially becomes basic Linux driver and application development, as would be done on a PC platform. Most of the application management is the same as PCs, for example, you run the same network daemons, use the same configuration

utilities. The embedded development kits from the Linux support houses like MontaVista are best at supporting these kinds of "canned" developments.

However, there is a second category of embedded systems. These systems do not rely on existing CPU cards. We can call this category "hard embedded". Typically, they are cost sensitive products like consumer products, which must justify every nickel of hardware cost. Memory size is still an issue. If you are designing a hard embedded system, Hollabaugh's book serves as a mere introduction. You will need to delve much deeper into the various Linux mechanisms to find ways to be more efficient. You will also need to consider thinner versions of various utilities, in particular you will need to learn to love busybox, the all in one executable. Support for loadable modules are probably tossed out; too much memory. Wrestling with library compatibilities will be a continuing issue.

If you are looking at a really low end product, you will need to look at the kernel. For example, uCLinux is the non-memory management version of Linux. It supports low end CPUs that do not have MMUs, or systems with very limited RAM space. However, many libraries and applications do not work on uCLinux and many features that you expect are either missing or do not work the same way as with standard kernel. None of this is addressed in the book.

Hollabaugh's writing style is excellent and his book says things that any embedded Linux developer should know. I highly recommend it. It would be nice if this book were the first of a trilogy. Volume 2 should cover additional subjects such as hard real time, networking (firewalls for embedded systems). Then Volume 3 should cover hard embedded systems that go beyond a simple canned SDK development.

Joel Williams
Mr. Williams operates his development consultancy under the name emLinux. Joel has been developing embedded Linux systems since 1996. He can be reached at joel@emLinux.com

---

By Dr. Ian McLoughlin

"Embedded Linux…" is a 419-page book describing the design and development of embedded linux systems from the point of view of a realistic multi-part example system: Project Trailblazer.

Trailblazer is an automation project for the Silverjack Resort, a winter resort located "somewhere in the Rocky Mountains." The automation covers data acquisition from temperature sensors, smart tag access control, image capture and display, audio playback, snow-making control, lift monitoring, race timing and one or two bonus side projects thrown in for good measure.

The content benefits from the examples chosen, and does not lose generality. Indeed the book covers an expansive territory encompassing interfacing electronics, device drivers, application software, integration code through to operating system choice and top-level design. The author somehow manages to deal with these topics for not just a single system but for 3 embedded platforms (StrongARM, x86 and PowerPC) and a desktop development platform.

Reading this book is like taking a ride down one of Project Trailblazers lower ski slopes. The going is gentle and smooth at first but soon gathers pace - you don't lose control, but at the bottom you turn around and are amazed at the territory you covered.

The gentle start includes the obligatory introductions and top-level design before moving on to system design considerations. These preliminary Chapters are no longer than necessary, but contain some no-nonsense information. In particular the 'Implications of Open Source' section covers legal aspects of the GPL and LGPL that many engineers would benefit from in discussions with management.

Luckily for us the Silverjack engineers could not agree in Chapter 2 on a single development platform, hence the StrongARM, x86 and PowerPC. These three threads are carried through the book, with every subsequent topic covered for each of the target architectures. There is also a wealth of background information on the platforms distributed throughout the book.

After a diversion in Chapter 3 to set up the x86-based development and cross-compilation system, we boot up the embedded systems in Chapter 4. Here I find my first complaint. These two chapters do contain a lot of explanation, but they also reprint some overly extensive scripts: listing 4.1 is almost 13 pages long (out of the chapter's 55 pages)! It's commented but not explained. Since the book has a companion website that carries the source code, it would have been better to drop the listing and insert an explanation instead. It's also a shame that a homegrown root file system wasn't fully developed, rather than use MontaVista's Journeyman. Journeyman is good - probably better than DIY (do it yourself) - but sometimes it's good to learn how to make your own.

A clear and concise treatment of local and remote debugging is next up in Chapter 5 followed by five chapters that really get to the heart of embedded systems. These cover asynchronous and synchronous serial communications as well as parallel port, USB and memory interfacing. User land control is introduced in each case, and hardware block diagrams are shown where appropriate, for which the author has kindly recommended a number of items of hardware, and frequently priced these. Where appropriate hardware tips are given, and oscilloscope plots too. The value of these cannot be overstated to engineers using this for a real project. It also demonstrates that the author 'practices what he preaches' and has done this for real.

Custom device drivers are described briefly but well. This text would get you up and running faster than the classic Rubini "Linux Device Drivers". The examples are up to date (kernel 2.4) and mostly use procfs interfaces. These chapters alone are reason enough to buy the book.

As a bonus, timing issues are covered in Chapter 11, along with latency tests using the standard Linux kernel. It turns out that latency is acceptable and real-time Linux kernel variants are not necessary for the Project Trailblazer. Real time Linux is mentioned only briefly, a potential downside since it would impact substantially on the solutions presented in some of the earlier chapters.

Finally, system integration is covered within the 30 pages of Chapter 12. A few too many listings again, but they are well explained in the text. There's nothing in Chapter 12 of the 23-hour days many engineers suffer during a system integration phase - by contrast it is all smooth sailing. Perhaps the author is showing his experience here?

The book resounds with the competence of a deeply experienced and knowledgeable author. There is no doubt that he knows his subject well, and communicates it in a readable way. The use of Project Trailblazer lends further realism to the chosen examples and ties them together in a coherent and user-friendly fashion. The book is a little long for cover-to-cover reading, but anyone embarking on a serious embedded linux project would undoubtedly benefit from the effort. This applies equally to experienced linux hackers migrating to embedded systems as well as to those new to linux.

This is a book to buy and use: it will be on the workbench more often than on the bookshelf.

Ian McLoughlin, Group Research
558 Wairakei Road, PO Box 1645
Burnside, Christchurch, New Zealand
ian.mcloughlin@tait.co.nz