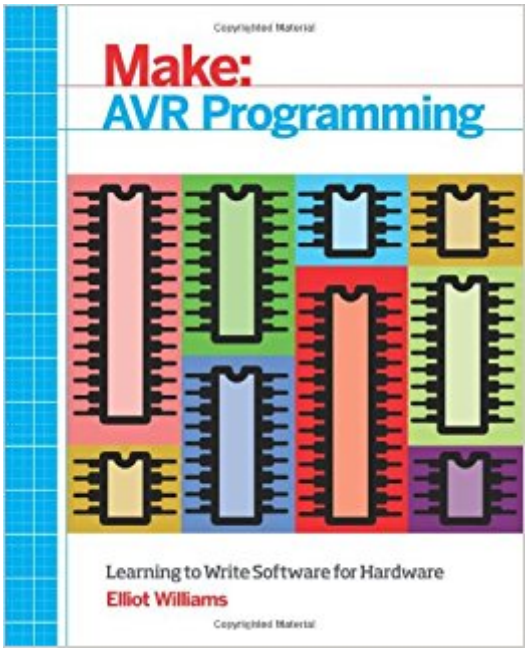
AVR Programming: Learning To Write Software For Hardware



Synopsis

Atmel’s AVR microcontrollers are the chips that power Arduino, and are the go-to chip for many hobbyist and hardware hacking projects. In this book you’ll set aside the layers of abstraction provided by the Arduino environment and learn how to program AVR microcontrollers directly. In doing so, you’ll get closer to the chip and you’ll be able to squeeze more power and features out of it.Each chapter of this book is centered around projects that incorporate that particular microcontroller topic. Each project includes schematics, code, and illustrations of a working project. Program a range of AVR chipsExtend and re-use other peopleÃ¢â ¬â„¢s code and circuitsInterface with USB, I2C, and SPI peripheral devicesLearn to access the full range of power and speed of the microcontrollerBuild projects including Cylon Eyes, a Square-Wave Organ, an AM Radio, a Passive Light-Sensor Alarm, Temperature Logger, and moreUnderstand what’s happening behind the scenes even when using the Arduino IDE

Book Information

Paperback: 474 pages

Publisher: Maker Media, Inc; 1 edition (February 17, 2014)

Language: English

ISBN-10: 1449355781

ISBN-13: 978-1449355784

Product Dimensions: 7.5 x 1 x 9.2 inches

Shipping Weight: 1.8 pounds (View shipping rates and policies)

Average Customer Review: 4.5 out of 5 stars 74 customer reviews

Best Sellers Rank: #154,605 in Books (See Top 100 in Books) #21 inÃ Â Books > Engineering & Transportation > Engineering > Electrical & Electronics > Electronics > Sensors #22 inÃ Â Books > Engineering & Transportation > Engineering > Electrical & Electronics > Circuits > Integrated #34 inÃ Â Books > Engineering & Transportation > Engineering > Electrical & Electronics > Electronics > Semiconductors

Customer Reviews

Unlock the full range of power and speed of Atmel’s chips

Elliot is a Ph.D. in Economics, a former government statistician, and a lifelong electronics hacker. He was among the founding members of HacDC, Washington DC’s hackerspace, and served as president and vice president for three years. He now lives in Munich, Germany, where he works for

an embedded hardware development firm that has, to date, exactly one employee (and CEO). This book came out of his experiences teaching AVR programming workshops at HacDC.

Excellent for getting started programming AVR MCU’s.For someone like myself with no microcontroller experience, some general electronics knowledge and lots of programming experience, this book hit the spot. A couple of pointers: - I got the USBTiny programmer from Sparkfun, which works fine with ATmega168’s. Just jumper the corresponding pins.- I started with the Arduino IDE, which works as mentioned in the book, but requires some fiddling with boards.txt to get USBTiny to work with atmega’s. In the end it all turned out to be more trouble than it’s worth. How it builds and flashes is pretty involved and far from transparent. Better to go with WinAVR as mentioned in the book, and just edit your code with good old Emacs. The Arduino IDE gets all wrapped around the axle if you’re messing with different processor speeds and baud rates. I never got the USART working right at anything but 9600 baud at 1MHz using the Arduino IDE. It’s much easier to just edit the Makefiles, which are very well documented. Also, if you’re messing with different AVR chips, you’ll want to go this route: the ATTiny chips don’t have a USART and the USART.c program doesn’t compile for them. You just remove that from the Makefile for ATTiny projects and you’re good-to-go. I have no idea what the other reviewer was talking about with WinAVR not working when the Arduino IDE was previously installed on a Windows PC - that was exactly my setup and it worked right away, subject to the following nit:- the avrdude argument -p should be "m168" not "m168p" as stated.- I ran through most of the examples to get going - they pretty much worked as expected.- When I started messing with the ADC, I realized how slow the internal one was and went for a separate chip. The MCP3004 works a lot faster and is fairly easy (and instructive) to interface via SPI. Again, the book does a great job at explaining SPI, and that knowledge translated well.- Another point to note is that some of the projects don’t work right if you’re connected to the programmer. Rather than plugging and unplugging all the jumpers each time, I attached the programmer to a breadboard with a ZIF socket, and had a separate breadboard and ZIF socket for running, and just shuttled the chip between the two.- Sometimes the book only shows a photo of the breadboard and not the actual wiring diagram. If you have to pick one, the wiring diagram is more clear IMO.I started this about 3-4 weeks ago from ground zero and have gotten through almost the entire book. At this point I feel very confident about working with AVR MCU’s going forward.

I like this book but for different reasons then "omg it’s so great"...I like it because it provides a

positive primer while not going to such detail that some theoretical person can read it nodding their head all the time.As a Primer, I would award this 5 stars but as a shelf resource I would give this 3.5 (hence the 4 start rating. I have noticed that the wording is not well edited and thus you have to re-read and fact check certain paragraphs. The code relies on you knowing and understanding libraries well. If you do not, get ready for some homework (which lets face it, you are programming an AVR, you HAVE TO KNOW libraries. You will also be required to understand AVR register programming such as the book does not teach well. Reference datasheets for the chip you are using. Ex: [search for the Atmel 328P COMPLETE datasheet (not the summary) on google] Also,...The book is consistently inconsistent. The author even notes in a few places that this is done on purpose to force a reader into studying outside material. Google comes in handy here and it is not a deal breaker. Again, Elliot is pushing you to use the internet instead of spoon feeding you literally everything. If you have determination and staying power and know how to work through these types of books and speedbumps found there-in, then go get it. If you are looking for a primer that will introduce you to the lingo and various use subjects, go get it. If you expect this to be a single source reference, think again!

I was pretty surprised to find that some people struggled with this book and rated it, what seems, unreasonably low to me. As I've been thinking this through, I've come to some conclusions as to why there might be such a huge gap between those experiences and my own. In a nutshell, I think it has to do with expectations going in. I have personally been able to be successful implementing the projects without too much trouble, but I went into it already being a seasoned software engineer and for any gaps that the book leaves out for specific implementation, I've been able to fill in with other tutorials online. My main filling in the gap experience comes from youtube--ymmv.You could make the case that a book should only earn a five star review if there were no need to fill in those gaps. I can understand that argument and even agree with it to some extent, however, the gaps in my case were more because I was using a different ISP to program the AVR than he uses and I was also using different chips. To me, though, that's my own problem. If you don't use exactly what the author prescribes for achieving success, you can hardly blame him because you can't make things work doing it your way. Even if you are coming at it with that view point, though, you simply can't deny that the book is very well written and Elliot's communication is concise and clear and not heady at all. He's really down to earth in his explanations. The book covers all the topics you'd want to know about when learning to program the AVR. I really can't think of anything he left out. And nowhere does the book claim to be a beginner book. He's not teaching C programming per se,

however, his explanations and hand holding with bit twiddling and the like are brilliant and not in the slightest bit patronizing. That part of the book alone is incredibly valuable.Anyhow, I give this book five stars because there is so much helpful detail and explanation around everything. You walk away from it understanding not only how things work, but you almost always know why as well. Why, for example, doesn't a servo rotate continuously like a DC motor? Because it has a different function. You'd use it for a joint on a robot's arm that should have a fixed range of motion rather than the perpetual turning you would need for a propeller motor. That type of explanation is plentiful throughout the book. The sidebars are always pertinent and help you think through what you're currently studying. The bottom line is that while your experience in putting together the project on your workbench may be varied, you cannot possibly argue that the material is not excellent. It's top notch. And if your complaint is that the github code is no good, that really should have no bearing on the way you evaluate the book itself, in my opinion. Maybe it makes the difference between a 5 star and a 4 star for some, but it most definitely should not reduce your rating to a 1 or 2 star. That's nonsense. The book's content is still really, really good.This book is terrific. If you're struggling with it, keep pushing on through, or maybe just keep doing Arduino programming until you feel more confident getting closer to the metal. Working there is fun. I'm confident this book can get you there, but it also will require some determination on your part.

to Software and Hardware Interfacing Computer Organization and Design, Fourth Edition: The Hardware/Software Interface (The Morgan Kaufmann Series in Computer Architecture and Design) The Architecture of Computer Hardware, Systems Software, & Networking: An Information Technology Approach IEC 61511-1 Ed. 1.0 b:2003, Functional safety - Safety instrumented systems for the process industry sector - Part 1: Framework, definitions, system, hardware and software requirements Specifying Systems: The TLA+ Language and Tools for Hardware and Software Engineers PIC Microcontroller: An Introduction to Software & Hardware Interfacing C++ and Python Programming: 2 Manuscript Bundle: Introductory Beginners Guide to Learn C++ Programming and Python Programming C++ and Python Programming 2 Bundle Manuscript. Introductory Beginners Guide to Learn C++ Programming and Python Programming Python Programming: The Complete Step By Step Guide to Master Python Programming and Start Coding Today! (Computer Programming Book 4)