

Hardware/Firmware Interface Design

*Best Practices for Improving
Embedded Systems Development*

Gary Stringham



ELSEVIER

AMSTERDAM • BOSTON • HEIDELBERG • LONDON
NEW YORK • OXFORD • PARIS • SAN DIEGO
SAN FRANCISCO • SINGAPORE • SYDNEY • TOKYO

Newnes is an imprint of Elsevier



Newnes

Newnes is an imprint of Elsevier
30 Corporate Drive, Suite 400, Burlington, MA 01803, USA
The Boulevard, Langford Lane, Kidlington, Oxford, OX5 1GB, UK

Copyright © 2010 by Elsevier Inc. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or any information storage and retrieval system, without permission in writing from the publisher. Details on how to seek permission, further information about the Publisher's permissions policies and our arrangements with organizations such as the Copyright Clearance Center and the Copyright Licensing Agency, can be found at our website: www.elsevier.com/permissions.

This book and the individual contributions contained in it are protected under copyright by the Publisher (other than as may be noted herein).

Notices

Knowledge and best practice in this field are constantly changing. As new research and experience broaden our understanding, changes in research methods, professional practices, or medical treatment may become necessary.

Practitioners and researchers must always rely on their own experience and knowledge in evaluating and using any information, methods, compounds, or experiments described herein. In using such information or methods they should be mindful of their own safety and the safety of others, including parties for whom they have a professional responsibility.

To the fullest extent of the law, neither the Publisher nor the authors, contributors, or editors, assume any liability 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.

Library of Congress Cataloging-in-Publication Data

Application Submitted.

British Library Cataloguing-in-Publication Data

A catalogue record for this book is available from the British Library.

ISBN: 978-1-85617-605-7

For information on all Newnes publications
visit our Web site at www.elsevierdirect.com

Typeset by diacriTech, Chennai, India

Printed in the United States of America

08 09 10 10 9 8 7 6 5 4 3 2 1

Working together to grow
libraries in developing countries

www.elsevier.com | www.bookaid.org | www.sabre.org

ELSEVIER

BOOK AID
International

Sabre Foundation

Contents

Preface	x
Chapter 1: Introduction	1
1.1. What Is the Hardware/Firmware Interface?	2
1.1.1. What Are Hardware, Chips, and Blocks?	2
1.1.2. What Are Firmware and Device Drivers?	6
1.2. What Is a Best Practice?	7
1.2.1. What Is a Principle?	9
1.2.2. Benefits of Principles and Practices	10
1.3. “First Time Right” Also Means	10
1.3.1. Easier to Program	11
1.3.2. Easier to Debug	11
1.3.3. Easier to Work around Defects	12
1.4. Target Audience	13
1.4.1. Hardware Engineers	13
1.4.2. Firmware Engineers	13
1.4.3. This Book in a University Setting	14
1.5. Project Life Cycle	14
1.6. Case Study	15
1.6.1. Monochrome Video Block in the Unity ASIC	15
1.6.2. A Case Study of a Good Example?	17
1.7. Summary	17
References	18
Chapter 2: Principles	19
2.1. Seven Principles of Hardware/Firmware Interface Design	19
2.1.1. Collaborate on the Design	20
2.1.2. Set and Adhere to Standards	21
2.1.3. Balance the Load	23
2.1.4. Design for Compatibility	25
2.1.5. Anticipate the Impacts	26
2.1.6. Design for Contingencies	27
2.1.7. Plan Ahead	29
2.2. Summary	30

Chapter 3: Collaboration	31
3.1. First Steps	31
3.1.1. Roles	31
3.1.2. Kick-off Activities	34
3.2. Formal Collaboration	35
3.2.1. Regular Meetings	35
3.2.2. Initial Firmware Support	37
3.2.3. Co-Development Techniques	38
3.2.4. End-Game Hardware Support	40
3.2.5. Documentation	41
3.3. Informal Collaboration	44
3.3.1. Formal Organizational Structure	44
3.3.2. Hardware Engineers' Initiative	45
3.3.3. Firmware Engineers' Initiative	46
3.3.4. Collaborative Problem Solving	47
3.4. Summary	48
3.4.1. Supporting Principles	49
References	49
Chapter 4: Planning	51
4.1. Industry Standards	51
4.1.1. Existing Standards	52
4.1.2. Implementing the Standard	53
4.1.3. Derivations or New Creations	55
4.2. Common Version	56
4.3. Compatibility	58
4.3.1. Range of Backward and Forward Compatibility	59
4.3.2. Combinations of Old vs. New	60
4.4. Defects	61
4.4.1. Document Defects	61
4.4.2. Fix Defects	63
4.4.3. Test Plan to Look for Defects	65
4.5. Analysis	66
4.5.1. Shared Pins	66
4.5.2. Buffer Management	67
4.5.3. Hardware/Firmware Interactions	67
4.5.4. Analyzing Third-Party IP	69
4.6. Postmortem	70
4.7. Summary	71
4.7.1. Supporting Principles	72
Chapter 5: Documentation	73
5.1. Types	74
5.1.1. Level and Types of Documentation	74
5.1.2. Chip-Level vs. Block-Level Documentation	75
5.1.3. Supported vs. Unsupported Documentation	77

5.2. Document Management	79
5.2.1. Document Standards	79
5.2.2. When to Write	80
5.2.3. Accuracy	81
5.3. Reviews	83
5.3.1. When to Review	83
5.3.2. Tracking Documentation Changes	84
5.3.3. Firmware Engineers' Responsibilities Regarding Reviews	85
5.4. Content	87
5.4.1. General Content	87
5.4.2. Sample Document Template	88
5.4.3. History	89
5.4.4. Features and Assumptions	91
5.4.5. Reference and Tutorial	92
5.4.6. Glossary and Errata	94
5.5. Registers	95
5.5.1. Document Registers	95
5.5.2. Register Design Tools	96
5.5.3. Table of Registers	100
5.5.4. Register Details and Description	101
5.6. Bits	103
5.6.1. Register Map Format	103
5.6.2. Bit Positions, Types, and Defaults	104
5.6.3. Bit Descriptions	106
5.6.4. Abort Impact	106
5.6.5. Test and Debug Bits	107
5.7. Interrupts	108
5.7.1. Edge- vs. Level-Triggered	108
5.7.2. Enabling and Acknowledging Interrupts	109
5.7.3. Interrupts Not Quite Done	110
5.7.4. Interrupts Repeating without Intervention	110
5.8. Time	111
5.8.1. Ranges of Time	111
5.8.2. Unit of Time	113
5.9. Errors	114
5.9.1. Two Types of Errors	115
5.9.2. Copious Information about the Errors	116
5.9.3. State of the Block after an Error	117
5.9.4. Firmware Steps to Recover	118
5.10. Information	119
5.10.1. Illegal Configuration	119
5.10.2. State Machines	119
5.10.3. How to Abort	120
5.11. Summary	121
5.11.1. Supporting Principles	122

Chapter 6: Superblock	123
6.1. Benefits of a Superblock	123
6.1.1. The Block's Entourage	124
6.1.2. Reasons for Having Unused Logic	124
6.1.3. Reasons against Having Unused Logic	129
6.2. Consolidation	131
6.2.1. Make a Superblock	131
6.2.2. Make a Supermodule	133
6.2.3. Evolutionary Design	133
6.2.4. Add Future Features	135
6.2.5. Superblock Version Number	137
6.3. I/O Signals	137
6.4. Parameterization	139
6.4.1. Reducing the Silicon Space	139
6.4.2. Minimizing Parameterization Risks	140
6.4.3. Parameterization Information for Firmware	142
6.4.4. Optional vs. Fixed Registers and Bits	145
6.5. Summary	146
6.5.1. Supporting Principles	147
Reference	147
Chapter 7: Design	149
7.1. Event Notification	149
7.1.1. No Indication	150
7.1.2. Timed Delay	151
7.1.3. Status Bit	153
7.1.4. Interrupts	155
7.2. Performance	157
7.2.1. Increasing the Buffer	158
7.2.2. Working Ahead	159
7.2.3. Tuning	160
7.2.4. Margins	161
7.3. Power-On	161
7.3.1. Power-On Interaction	161
7.3.2. Power-On State of I/O Lines	163
7.3.3. Block-Level Power Control	163
7.4. Communication and Control	164
7.4.1. Error Information	164
7.4.2. DMA Features	164
7.4.3. Sharing I/O Pins	166
7.4.4. Hiding Implementation Details	167
7.5. Summary	169
7.5.1. Supporting Principles	170

Chapter 8: Registers	171
8.1. Addressing	172
8.1.1. Processor Access	172
8.1.2. Chip Base Addresses	175
8.1.3. Block Offset and Base Addresses	176
8.1.4. Register Offset Addresses	178
8.1.5. Sub-Blocks	179
8.1.6. Bursting	179
8.1.7. Unused Address Locations	180
8.1.8. Changes in the Next Chip	181
8.2. Bit Assignment	183
8.2.1. Assigning Bit Positions	183
8.2.2. Multi-Bit Fields	185
8.2.3. Multi-Register Fields	187
8.2.4. Unused Bit Positions	188
8.2.5. Changes in the Next Revision	189
8.2.6. Bit Types	192
8.2.7. Bit Types in Registers	195
8.2.8. Grouping by Operational Mode	197
8.2.9. Multiple Instantiations of a Block	198
8.3. Data Types	199
8.3.1. Integers	200
8.3.2. Real Numbers	201
8.3.3. Pointers	205
8.3.4. Constants	207
8.4. Hardware Identification	207
8.4.1. Chip ID and Version	208
8.4.2. Block ID and Version	209
8.5. Communication and Control	210
8.5.1. Necessary Information	210
8.5.2. Queuing Tasks in the Block	211
8.5.3. Coherent Register Contents	216
8.5.4. Atomic Register Access	217
8.6. Summary	221
8.6.1. Supporting Principles	222
Chapter 9: Interrupts	223
9.1. Design	224
9.1.1. An Interrupt Supermodule	224
9.1.2. Hierarchical Interrupt Structure	226
9.1.3. Interrupt Sharing	228
9.1.4. Source Signal Integrity	230
9.1.5. Types of Interrupt Triggers	231

9.2. Pending Register	236
9.2.1. Acknowledging an Interrupt	236
9.2.2. Order of Interrupt Positions	239
9.3. Enable Register	240
9.3.1. A 1 Enables the Interrupt	241
9.3.2. Enable Controls Interrupt	241
9.3.3. Default Settings for Enable	243
9.4. Optional Registers	243
9.4.1. Source Status Register	243
9.4.2. Post Register	245
9.4.3. Atomic Enable/Disable Registers	245
9.4.4. Masked Register	246
9.4.5. Instantiation Register	246
9.4.6. Addresses of Optional Registers	247
9.5. Interrupt Module Review	248
9.5.1. Interrupt Channels	249
9.5.2. Interrupt Module	251
9.5.3. External Connections	252
9.6. Triggering on Both Edges	253
9.6.1. Use Two Interrupt Channels	253
9.6.2. Channel Positions of Leading and Trailing Interrupts	255
9.7. Using the Interrupt Module	257
9.7.1. When to Allocate an Interrupt Channel	257
9.7.2. Repeated Interrupts	259
9.7.3. Address Mapping	259
9.8. Summary	260
9.8.1. Supporting Principles	261
Chapter 10: Aborts, etc.	263
10.1. Definitions	263
10.2. Halts	265
10.3. Resets	266
10.4. Aborts	268
10.4.1. The Need for Aborts	268
10.4.2. Firmware’s Interaction with Aborts	270
10.4.3. Abort Behavior	272
10.4.4. Abort Interactions between Blocks	274
10.5. Summary	275
10.5.1. Supporting Principles	276
Chapter 11: Hooks	277
11.1. Designing for Hooks	278
11.1.1. What Hooks to Add	279
11.1.2. Adding Registers	279

11.1.3. Looking for Potential Problem Areas	280
11.1.4. Removing Workarounds	280
11.2. Peek	281
11.2.1. Internal Registers	281
11.2.2. Signals	282
11.2.3. Memory	283
11.2.4. State Machines	285
11.3. ... And Poke	287
11.3.1. Destructive Reads and Writes	287
11.3.2. Input and Output Signals	288
11.3.3. Overwriting Registers	289
11.4. Monitor	289
11.4.1. Event Tracking	289
11.4.2. Timers	291
11.4.3. Data Watching	292
11.5. More Hooks	293
11.5.1. Bypass Paths	293
11.5.2. Extra Resources for Test and Debug	295
11.5.3. Dedicated Processor	297
11.6. Summary	298
11.6.1. Supporting Principles	299
Chapter 12: Conclusion	301
12.1. Key Points	301
12.2. Benefits	302
12.3. Seven Principles of Hardware/Firmware Interface Design	302
12.4. It Finally Works! Let's Ship It!	303
Appendix A: Best Practices	307
Appendix B: Bicycle Controller Specification	327
Appendix C: elsevierdirect.com/companions/9781856176057 or garystringham.com/hwfwbook	
Appendix D: Glossary	345
Index	349

Preface

You can find books written by hardware engineers teaching hardware engineers how to design hardware. You can find books written by firmware engineers teaching firmware engineers how to write firmware. This book is written by a firmware engineer but is directed primarily to hardware engineers.

Many engineers have experienced problems when trying to get firmware working on hardware. They are designed generally in isolation from each other and then are expected to work when brought together. But problems and defects appear. At times it is unknown where the defect is located—in hardware or firmware, or maybe the documentation.

There is very little written about how to get hardware and firmware to work well together. This book attempts to fill that niche. It addresses the interface between the hardware and firmware domains and provides practices that will reduce the time and effort required to produce an embedded systems product. It covers all aspects of development surrounding the hardware/firmware interface, including the process of development, the high-level design, and the detailed design.

A key feature of this book are the 300+ Best Practices that give detailed instructions for various aspects of the development process and design. These best practices apply perfectly, but only for a given situation. They should be scrutinized for applicability in a given situation. Throughout this book, the emphasis is for engineers to develop their own set of best practices. They may start with these 300, but the set should evolve to be made their own, as this increases the likelihood of success within their organization.

To help engineers understand the 300+ Best Practices, and to help them create their own set, Seven Principles are presented that provide overarching guidelines and direction. These principles, when internalized, will help engineers work in the right direction, even if there is no specific best practice for that situation. Following the Seven Principles and 300+ Best Practices will improve the design teams' ability to produce successful embedded systems products.

Chapter Summaries

The following chapter summaries provide an overview of the book and help the reader to navigate through the book.

1. **Introduction:** This chapter establishes the foundation for the book. It discusses various types of hardware and how they impact the hardware/firmware interface. It defines principles and best practices, the target audience, and the product life cycle. It also presents a case study used throughout the book.
2. **Principles:** This chapter presents the Seven Principles and provides a high-level view and reasoning for the direction of this book. Understanding these principles is key to understanding why the best practices are stated as they are.
3. **Collaboration:** Of key importance to the success of an embedded product is the proper and sufficient collaboration between hardware and firmware engineers. This chapter defines roles and processes in such an effort.
4. **Planning:** Before starting a project, planning must be done to determine and agree what direction should be taken with the new product. This chapter covers several areas that should be visited when planning a new project.
5. **Documentation:** Most engineers assigned to write documentation do not like the task. And most engineers reading documentation get frustrated with incomplete and incorrect documentation. This chapter discusses the types of documentation, when to write them, how to review them, and what types of details to include in them.
6. **Superblock:** This chapter introduces the concept of a block that can do everything within its own domain. It discusses why a superblock is good and how to set it up to be used where needed. But it also discusses the reality of practical limitations and how to handle those.
7. **Design:** Various design aspects are discussed in this chapter, such as events, power-on sequences, communication, and control.
8. **Registers:** Registers are the fundamental interface between hardware and firmware. This chapter discusses them in great detail, including addresses, bit locations, and types of bits.
9. **Interrupts:** Given a lack of consistency among interrupt designs used in the industry, this chapter focuses in great detail how interrupts from hardware into firmware should be managed. This chapter also contains a proposal for an interrupt standard and discusses the proposal in detail.
10. **Aborts, etc.:** Too often very little thought is given to errors and how to recover from them. This chapter discusses design elements necessary to allow firmware to abort hardware operations, recover, and resume processing.

11. **Hooks:** Logic analyzers cannot probe the internals of a chip but knowledge of what is occurring inside is important when trying to get firmware working on hardware. Having firmware-accessible hooks inside the chip allows firmware to retrieve information for engineering analysis. This chapter contains many possible hooks that could be included.
12. **Conclusion:** This chapter wraps up the book. It also contains a couple of cartoon illustrations used to help illustrate the concepts in the book.

Appendices

- A. **Best Practices:** This appendix collects all the best practices in the book into one place.
- B. **Block Specification:** This appendix is a documentation template as explained and described in Chapter 5, Documentation.
- C. **Using This Book in a University:** This appendix provides suggestions on how to use this book to teach hardware and firmware engineering students that have to work together on a project.
- D. **Glossary:** Given that this book addresses two different engineering disciplines, hardware and firmware engineering, it covers terms from one domain that might not be understood by the other.

Conventions Used in This Book

The bulk of the text in this book discusses the concept at hand. Interspersed in the text are one or more of these elements: figures, listings, register maps, best practices, and tales from the trenches.

Figures

Figure 0.1 is an example figure.

Firmware Listings

Listing 0.1 shows an example listing of firmware source code written in C.

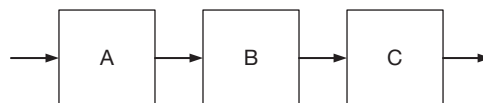


Figure 0.1: Example figure.

Listing 0.1: Example C code listing.

```
/* Read the current list of pending interrupts */
interrupts = *interruptRegister;
```

Hardware Circuits

A few hardware circuits are illustrated in the book. Both a schematic drawing and its equivalent Verilog listing will be given. Figure 0.2 is the schematic and Listing 0.2 is the corresponding Verilog code for an example circuit.

Register Maps

This diagrammatic form is used in discussions about registers, how various bits are mapped into the register, and the mode of operation and reset values of these bits. A detailed explanation of this format is given in Chapter 5, Documentation.

	Daily Register – 0x0004																LSB															
Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R/W	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	C	B	A
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- A This bit means one thing.
- B This bit means another.
- C And this bit means something else.

Best Practices

The book contains 300+ Best Practices related to the concepts being taught. In addition to presentation throughout the book, these practices are collected in Appendix A, thereby



Figure 0.2: Schematic for example circuit.

Listing 0.2: Verilog code for example circuit.

```
// A simple AND gate.
assign c = a & b;
```

providing a concise checklist that can be used during chip design projects. They are also provided in a spreadsheet available online at the publisher's website, elsevierdirect.com/companions, and at the author's website, garystringham.com/hwfwbook.

Each best practice has an ID number, X.Y.Z, which is used in the body of the book, in Appendix A, and in the spreadsheet.

Best Practice

1.1.1 Best Practices of Hardware/Firmware Interface Design.

Like the book, the Excel spreadsheet database is copyrighted material. Purchasers of this book are entitled (and encouraged) to start with the database and modify it to suit the needs of their design team, but some restrictions apply. See Appendix A for more details on the database and its copyright permissions.

Best Practice

1.1.2 Copyright © 2009, Gary Stringham & Associates, LLC. All rights reserved. Do not distribute beyond your team.

Tales from the Trenches

Scattered throughout this book are real-life stories that help illustrate the impact of the topic at hand. These are stories from real engineers (mostly me) in the trenches, working away designing and solving problems. The following is an example tale (not a real one).

 **Tales from the Trenches**

I remember hearing a story from a friend of a friend, who heard that an engineer had said that he heard a manager tell her subordinate that—according to the rumor she had heard—it was already broken to start with.

Companion Website

This book has a companion website at elsevierdirect.com/companions/9781856176057, where you will find links to the spreadsheet database for the 300+ Best Practices, the document template discussed in Chapter 5, Documentation, and other related content. Please

visit the author's website at garystringham.com/hwfwbook for the same tools, plus additional links to his work in this area and details of how to contact him directly.

How to Contact Me

If you have any questions about the content of this book or about your hardware/firmware interface design, feel free to contact me at gary@garystringham.com.


Acknowledgments

I would like to thank Jack Meador and Mike Merrell, the two unlucky hardware engineers who had to put up with my constant questions, issues, and requests as we worked through the project that was the catalyst for this book. They provided valuable insight and help from within their hardware domain. They, along with other hardware and firmware engineers within the organization and from other companies, provided much of the input used in many of the best practices and tales from the trenches in this book.

I would also like to thank my immediate managers at the time, Warren Johnson and Tracy Sauerwein, and managers above them, Sandy Lieske and Von Hansen. The book is finally published—your unwarranted support, while tracing my progress from sandy to smooth, was not in vain.

My badly English writing was greatly improved through the patient tutelage of my technical writing coach, Joel Saks. He has a gift with words that is way beyond my abilities. In addition, he was a valuable resource for critical analysis of my concepts, pushing me to clearly articulate and justify what seemed obvious to me.

I would also like to thank John Blyler, Clive “Max” Maxfield, Jack Meador, Mike Merrell, Joel Saks, and three others (who wish to remain anonymous) for reviewing all or parts of my book. Your comments provided valuable input and suggestions, making the book better than otherwise. Thanks to Mike Merrell for his help with the Verilog code and to Kevin Falk for drawing the car illustrations. And thanks to the many others who have given me suggestions and enthusiastic encouragement during the 5 years it took me to complete this project.

Most of all, I want to thank my wife and children  for their patience and long suffering as I spent evenings and weekends working on this book instead of making repairs on the house, driving the children to their activities, and vacationing with the family.