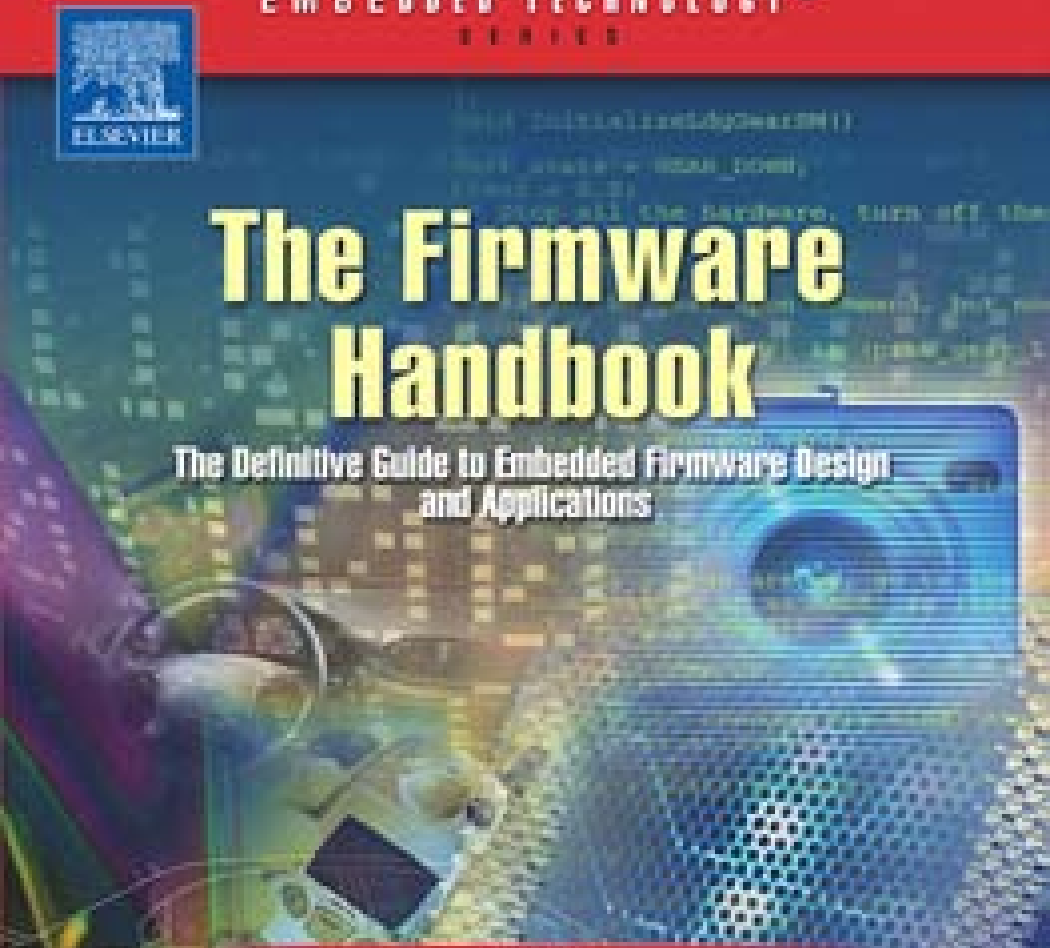
EMBEDDED TECHNOLOGY™
SERIES



The Firmware Handbook

The Definitive Guide to Embedded Firmware Design and Applications

Edited by

Jack Ganssle

CD-ROM included
Contains source code, an exclusive Firmware Standards article by Jack Ganssle, and a searchable eBook version of the book

Newnes

THE FIRMWARE HANDBOOK

THE FIRMWARE HANDBOOK

Edited by

Jack Ganssle

ELSEVIER

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

Newnes is an imprint of Elsevier

Newnes is an imprint of Elsevier
200 Wheeler Road, Burlington, MA 01803, USA
Linacre House, Jordan Hill, Oxford OX2 8DP, UK

Recognizing the importance of preserving what has been written, Elsevier prints its books on acid-free paper whenever possible.

Library of Congress Cataloging-in-Publication Data

Ganssle, Jack
The firmware handbook / by Jack Ganssle.
p. cm.
ISBN 0-7506-7606-X
1. Computer firmware. I. Title.

QA76.765.G36 2004
004—dc22 2004040238

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library.

For information on all Newnes publications
visit our website at www.newnespress.com

04 05 06 07 08 10 9 8 7 6 5 4 3 2 1

Printed in the United States of America

Acknowledgements

I'd like to thank all of the authors whose material appears in this volume. It's been a big job, folks, but we're finally done!

Thanks also to Newnes acquisition editor Carol Lewis, and her husband Jack, who pestered me for a year before inventing the book's concept, in a form I felt that made sense for the readers and the authors. Carol started my book-writing career over a decade ago when she solicited *The Art of Programming Embedded Systems*. Thanks for sticking by me over the years.

And especially, thanks to my wife, Marybeth, for her support and encouragement, as well as an awful lot of work reformatting materials and handling the administrivia of the project.

—Jack Ganssle, February 2004, Anchorage Marina, Baltimore, MD

Contents

Preface

Welcome to The Firmware Handbook! This book fills a critical gap in the embedded literature. There's a crying need for a compendium of ideas and concepts, a handbook, a volume that lives on engineers' desks, one they consult for solutions to problems and for insight into forgotten areas. Though the main topic is firmware, the gritty reality of the embedded world is that the code and hardware are codependent. Neither exists in isolation; in no other field of software is there such a deep coupling of the real and the virtual.

Analog designers tout the fun of their profession. Sure, it's a gas to slew an op amp. But those poor souls know nothing of the excitement of making things move, lights blink, gas flow. We embedded developers sequence motors, pump blood, actuate vehicles' brakes, control the horizontal and vertical on TV sets and eject CDs. Who can beat the fuzzy line between firmware and the real world for fun?

The target audience for the book is the practicing firmware developer, those of us who write the code that makes the technology of the 21st century work.

The book is not intended as a tutorial or introduction to writing firmware. Plenty of other volumes exist whose goal is teaching people the essentials of developing embedded systems.

Nor is it an introduction to the essentials of software engineering. Every developer should be familiar with Boehm's COCOMO software estimation model, the tenants of eXtreme Programming, Fagen and Gilb's approaches to software inspections, Humphrey's Personal Software Process, the Software Engineering Institute's Capability Maturity Model, and numerous other well-known and evolving methodologies. Meticulous software engineering is a critical part of success in building any large system, but there's little about it that's unique to the embedded world.

The reader also will find little about RTOSes, TCP/IP, DSP and other similar topics. These are hot topics, critical to an awful lot of embedded applications. Yet each subject needs an entire volume to itself, and many, many such books exist.

What's on the CD-Rom?

Included on the accompanying CD-ROM:

- A full searchable eBook version of the text in Adobe pdf format
- A directory containing the source code for all of the example programs in the book

Refer to the ReadMe file for more details on CD-ROM content.

SECTION

I

Basic Hardware

Introduction

The earliest electronic computers were analog machines, really nothing more than a collection of operational amplifiers. "Programs" as we know them did not exist; instead, users created algorithms using arrays of electronic components placed into the amps' feedback loops. Programming literally meant rewiring the machine. Only electrical engineers understood enough of the circuitry to actually use these beasts.

In the 1940s the advent of the stored program digital computer transformed programs from circuits to bits saved on various media… though nothing that would look familiar today! A more subtle benefit was a layer of abstraction between the programmer and the machine. The digital nature of the machines transcended electrical parameters; nothing existed other than a zero or a one. Computing was opened to a huge group of people well beyond just electrical engineers. Programming became its own discipline, with its own skills, none of which required the slightest knowledge of electronics and circuits.

Except in embedded systems. Intel's 1971 invention of the microprocessor brought computers back to their roots. Suddenly computers cost little and were small enough to build into products. Despite cheap CPUs, though, keeping the overall system cost low became the new challenge for designers *and* programmers of these new smart products.

This is the last bastion of computing where the hardware and firmware together form a unified whole. It's often possible to reduce system costs by replacing components with smarter code, or to tune performance problems by taking the opposite tack. Our firmware works intimately with peripherals, sensors and a multitude of real-world phenomena. Though there's been a trend to hire software-only people to build firmware, the very best developers will always be those with a breadth of experience that encompasses the very best software development practices with a w orking knowledge of the electronics.

Every embedded developer should be—no, *must* be—familiar with the information in this chapter. You can't say "well, I'm a firmware engineer, so I don't need to know what a resistor is." You're an *embedded* engineer, tasked with building a system that's more than just code.

A fantastic reference that gives much more insight into every aspect of electronics is *The Art of Electronics* by Horowitz and Hill.