

GAME CONSOLE HACKING

Xbox, Playstation, Nintendo,
Atari, & Gamepark 32

- Learn the Basics of Game Console Hacking and Reverse Engineering
- Modify Video Game Consoles and Accessories To Do Things They Were Never Intended To Do
- Create and Play Your Own Games on Your Favorite Systems
- Featuring the Xbox, PlayStation 2, Game Boy Advance, GP32, NES, Atari 2600, Atari 5200, and Atari 7800

Joe Grand

**Frank Thornton
Albert Yarusso**

Special Foreword by
Ralph H. Baer
“The Father of Video Games”

Register for Free Membership to

solutions@syngress.com

Over the last few years, Syngress has published many best-selling and critically acclaimed books, including Tom Shinder's *Configuring ISA Server 2000*, Brian Caswell and Jay Beale's *Snort 2.0 Intrusion Detection*, and Angela Orebaugh and Gilbert Ramirez's *Ethereal Packet Sniffing*. One of the reasons for the success of these books has been our unique **solutions@syngress.com** program. Through this site, we've been able to provide readers a real time extension to the printed book.

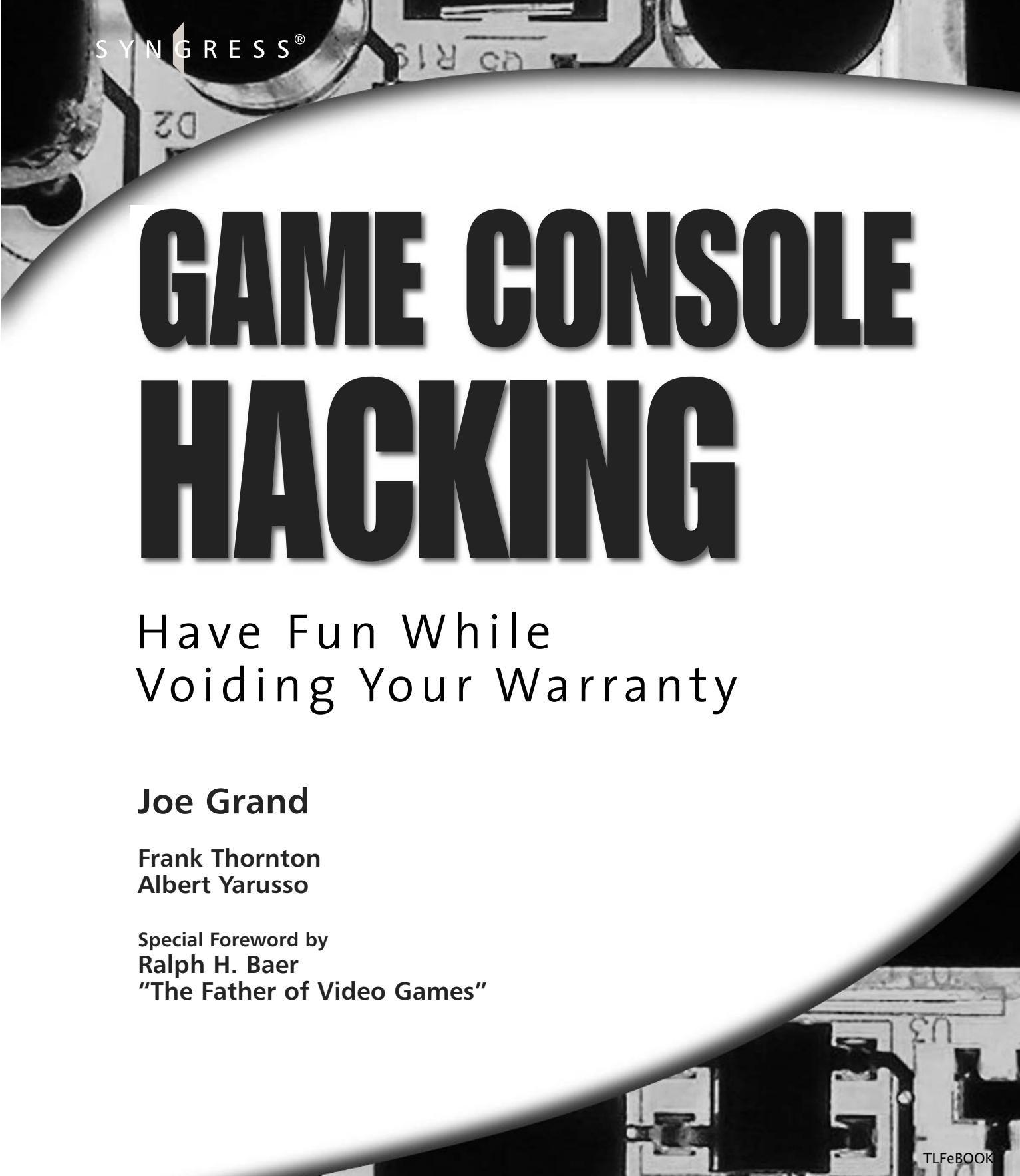
As a registered owner of this book, you will qualify for free access to our members-only **solutions@syngress.com** program. Once you have registered, you will enjoy several benefits, including:

- Four downloadable e-booklets on topics related to the book. Each booklet is approximately 20-30 pages in Adobe PDF format. They have been selected by our editors from other best-selling Syngress books as providing topic coverage that is directly related to the coverage in this book.
- A comprehensive FAQ page that consolidates all of the key points of this book into an easy to search web page, providing you with the concise, easy to access data you need to perform your job.
- A "From the Author" Forum that allows the authors of this book to post timely updates links to related sites, or additional topic coverage that may have been requested by readers.

Just visit us at **www.syngress.com/solutions** and follow the simple registration process. You will need to have this book with you when you register.

Thank you for giving us the opportunity to serve your needs. And be sure to let us know if there is anything else we can do to make your job easier.

S Y N G R E S S ®



SYNGRESS®

GAME CONSOLE HACKING

Have Fun While
Voiding Your Warranty

Joe Grand

Frank Thornton
Albert Yarusso

Special Foreword by
Ralph H. Baer
“The Father of Video Games”

Syngress Publishing, Inc., the author(s), and any person or firm involved in the writing, editing, or production (collectively “Makers”) of this book (“the Work”) do not guarantee or warrant the results to be obtained from the Work.

There is no guarantee of any kind, expressed or implied, regarding the Work or its contents. The Work is sold AS IS and WITHOUT WARRANTY. You may have other legal rights, which vary from state to state.

In no event will Makers be liable to you for damages, including any loss of profits, lost savings, or other incidental or consequential damages arising out from the Work or its contents. Because some states do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitation may not apply to you.

You should always use reasonable care, including backup and other appropriate precautions, when working with computers, networks, data, and files.

Syngress Media®, Syngress®, “Career Advancement Through Skill Enhancement®,” “Ask the Author UPDATE®,” and “Hack Proofing®,” are registered trademarks of Syngress Publishing, Inc. “Syngress: The Definition of a Serious Security Library”™, “Mission Critical™,” and “The Only Way to Stop a Hacker is to Think Like One™” are trademarks of Syngress Publishing, Inc. Brands and product names mentioned in this book are trademarks or service marks of their respective companies.

KEY SERIAL NUMBER

001	JKL32CVF79
002	P5FGJK9995
003	82H24555YY
004	38IIHGF543
005	CVPLQ6WQ23
006	VT5123HG66
007	H3WD3EHJNB
008	29WMKB8765
009	62SDJTHGGG
010	I5TBBB536T

PUBLISHED BY

Syngress Publishing, Inc.
800 Hingham Street
Rockland, MA 02370

Game Console Hacking: Xbox, Playstation, Nintendo, Atari, & Gamepark 32

Copyright © 2004 by Syngress Publishing, Inc. All rights reserved. Printed in the United States of America. Except as permitted under the Copyright Act of 1976, no part of this publication may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written permission of the publisher, with the exception that the program listings may be entered, stored, and executed in a computer system, but they may not be reproduced for publication.

Printed in the United States of America

1 2 3 4 5 6 7 8 9 0

ISBN: 1-931836-31-0

Publisher: Andrew Williams

Acquisitions Editor: Christine Kloiber

Technical Editor: Joe Grand

Cover Designer: Michael Kavish

Page Layout and Art: Patricia Lupien

Copy Editor: Darlene Bordwell

Indexer: J. Edmund Rush

Distributed by O'Reilly Media, Inc. in the United States and Canada.

For information on rights and translations, contact Matt Pedersen, Director of Sales and Rights, at Syngress Publishing; email matt@syngress.com or fax to 781-681-3585.



Acknowledgments

We would like to acknowledge the following people for their kindness and support in making this book possible.

Syngress books are now distributed in the United States and Canada by O'Reilly Media, Inc. The enthusiasm and work ethic at O'Reilly is incredible and we would like to thank everyone there for their time and efforts to bring Syngress books to market: Tim O'Reilly, Laura Baldwin, Mark Brokering, Mike Leonard, Donna Selenko, Bonnie Sheehan, Cindy Davis, Grant Kikkert, Opol Matsutaro, Steve Hazelwood, Mark Wilson, Rick Brown, Leslie Becker, Jill Lothrop, Tim Hinton, Kyle Hart, Sara Winge, C. J. Rayhill, Peter Pardo, Leslie Crandell, Valerie Dow, Regina Aggio, Pascal Honscher, Preston Paull, Susan Thompson, Bruce Stewart, Laura Schmier, Sue Willing, Mark Jacobsen, Betsy Waliszewski, Dawn Mann, Kathryn Barrett, John Chodacki, and Rob Bullington.

The incredibly hard working team at Elsevier Science, including Jonathan Bunkell, Ian Seager, Duncan Enright, David Burton, Rosanna Ramacciotti, Robert Fairbrother, Miguel Sanchez, Klaus Beran, Emma Wyatt, Rosie Moss, Chris Hossack, Mark Hunt, and Krista Leppiko, for making certain that our vision remains worldwide in scope.

David Buckland, Marie Chieng, Lucy Chong, Leslie Lim, Audrey Gan, Pang Ai Hua, and Joseph Chan of STP Distributors for the enthusiasm with which they receive our books.

Kwon Sung June at Acorn Publishing for his support.

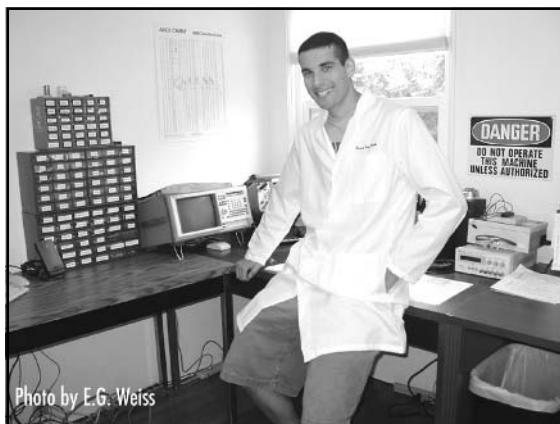
David Scott, Tricia Wilden, Marilla Burgess, Annette Scott, Andrew Swaffer, Stephen O'Donoghue, Bec Lowe, and Mark Langley of Woodslane for distributing our books throughout Australia, New Zealand, Papua New Guinea, Fiji Tonga, Solomon Islands, and the Cook Islands.

Winston Lim of Global Publishing for his help and support with distribution of Syngress books in the Philippines.

A special thank you to our attorney and friend Gene Landy, whose expertise in "all things intellectual property" is impressive.



Technical Editor & Contributor



Joe Grand; Grand Idea Studio, Inc. Joe Grand is the President of Grand Idea Studio, a San Diego-based product development and intellectual property licensing firm, where he specializes in the invention and design of consumer electronics, medical devices, video games, and toys. His latest creations include the Stelladaptor Atari 2600 Controller-to-USB Interface and the Emic Text-to-Speech Module.

A recognized figure in computer security, Joe has testified before the United States Senate Governmental Affairs Committee and is a former member of the legendary hacker collective L0pht Heavy Industries. Joe's research on mobile devices and embedded security has been published in various periodicals, including *Circuit Cellar* and the *Digital Investigation Journal*. He is the author of many security-related software tools, including pdd, the first forensic acquisition application for Palm devices. Joe currently has a patent pending on a hardware-based computer memory imaging concept and apparatus (U.S. Patent Serial No. 10/325,506).

Joe has presented his work at numerous academic, industry, and private forums, including the United States Air Force Office of Special Investigations, the Naval Postgraduate School, the IBM Thomas J. Watson Research Center, the Embedded Systems Conference, the Black Hat Briefings, and DEFCON. He has appeared in documentaries and news for television, airplane in-flight programming, and print media outlets. He has also authored *Hardware Hacking: Have Fun While Voiding Your Warranty* (Syngress Publishing, ISBN: 1-932266-83-6), contributed to *Stealing The Network: How to Own A Continent* (Syngress, ISBN: 1-931836-05-1), and is a frequent contributor to other texts. Joe holds a Bachelor of Science degree in Computer Engineering from Boston University.

Joe is the author of Chapter 1 “Tools of the Warranty Voiding Trade,” Chapter 2 “Case Modifications: Building an Atari 2600PC,” Chapter 5 “Nintendo GBA,” Chapter 6 “GP32,” Chapter 7 “NES,” and the Appendices.

Contributors

Frank (Thorn) Thornton runs his own technology-consulting firm, Blackthorn Systems, which specializes in wireless networks. His specialties include wireless network architecture, design, and implementation, as well as network troubleshooting and optimization. An interest in amateur radio has also helped him bridge the gap between computers and wireless networks. Frank's experience with computers goes back to the 1970's when he started programming mainframes. Over the last 30 years, he has used dozens of different operating systems and programming languages. Having learned at a young age which end of the soldering iron was hot, he has even been known to repair hardware on occasion. In addition to his computer and wireless interests, Frank was a law enforcement officer for many years. As a detective and forensics expert he has investigated approximately one hundred homicides and thousands of other crime scenes. Combining both professional interests, he was a member of the workgroup that established ANSI Standard *ANSI/NIST-CSL 1-1993 Data Format for the Interchange of Fingerprint Information*. He has co-authored *WarDriving: Drive, Detect, and Defend: A Guide to Wireless Security* (Syngress Publishing, ISBN: 1-93183-60-3), as well as contributed to *IT Ethics Handbook: Right and Wrong for IT Professionals* (Syngress, ISBN: 1-931836-14-0). He resides in Vermont with his wife.

Frank is the author of Chapter 3 "Xbox."

Albert Yarusso is a principle of Austin Systems (www.austinsystems.com), an Austin, Texas-based firm that specializes in web design programming and hosting services. Albert's background consists of a wide range of projects as a software developer, with his most recent experience focused in the game industry. Albert previously worked for Looking Glass Technologies and more recently for Ion Storm Austin, where he helped create the highly acclaimed PC game Deus Ex.

Albert co-founded AtariAge (www.atariage.com) in 2001, a comprehensive website devoted to preserving the history of Atari's rich legacy of video game consoles and computers, which has become one of the busiest destinations on the web for classic gaming fans. In 2003, Albert helped bring the first annual Austin Gaming Expo (www.austingamingexpo.com) to Austin, an extremely successful event that drew over 2,000 visitors in its first year. Albert is also a contributor to *Hardware Hacking: Have Fun While Voiding Your Warranty* (Syngress Publishing, ISBN: 1-932266-83-6).

Albert is the author of Chapter 8 “Atari 2600,” Chapter 9 “Atari 5200 SuperSystem,” and Chapter 10 “Atari 7800.”

Jonathan S. Harbour has been an avid hacker for many years, having started with early systems like the Commodore PET, Apple II, and Tandy 1000. He holds a degree in computer information systems, enjoys writing code in C, C++, and several other languages, and has experience with many platforms, including Windows, Linux, Pocket PC, and Game Boy Advance. Jonathan has written several books on the subject of game programming, and may be contacted via his Web site at www.jharbour.com.

Jonathan is a contributor to Chapter 5 “Nintendo GBA.”

Marcus R. Brown is a software engineer at Budcat Creations. His work includes writing low-level drivers and system-level programming such as resource management, file loading, and audio streaming. He is currently working on an unannounced title for the PlayStation 2 and Xbox. Marcus lives in Las Vegas, Nevada.

Marcus is the author of Chapter 4 “PlayStation 2.”

Christopher Dolberg is a full-time student, and an avid player of console and PC games. When not gaming, he can be found modifying his hardware in an attempt to push it to the very limits of its function. Occasionally he takes time off from both these activities to actually attend classes. He resides in Vermont.

Chris is a contributor to Chapter 3 “Xbox.”



Foreword Contributor

Ralph H. Baer is an engineer and a hacker from way back, as well as a prolific inventor with over 150 US and foreign patents to his credit. He is best known as the “Father of Video Games.” For over fifty years he has had one leg in the commercial and defense electronics development and production business; and the other leg in toy and game design. Many well-known handheld electronic toys such as “Simon” came from his lab. His early video game hardware already resides in such places as the Smithsonian and the Japanese National Science Museum and replicas are on display all over the map.

His home has been Manchester, New Hampshire for the past 48 years. He moves around a lot.



Technical Reviewer

Job de Haas is Managing Director of ITSX BV, a Dutch company located in Amsterdam. ITSX BV provides security testing services in the broadest sense. Job is involved in testing, researching, and breaking security aspects of the latest technologies for corporate clients. In assignments for telecommunication operators and mobile phone manufacturers, Job gained experience with internal operations of modern phones.

Job holds a master’s degree in electrical engineering from Delft Technical University. He previously held positions at the Dutch Aerospace Agency (NLR) as a robotics researcher and at Digicash BV as a developer of cryptographic applications. He lives in Amsterdam, The Netherlands.

Contents

Forewordxxi
Introduction 2.0xxvii
Introductionxxix
Part I Introduction to Hardware Hacking1
Chapter 1 Tools of the Warranty-Voiding Trade3
Introduction4
The Essential Tools5
Basic Hardware Hacking8
Advanced Projects and Reverse Engineering13
Where to Obtain the Tools16
Chapter 2 Case Modifications: Building an Atari 2600PC19
Introduction20
Choosing Your Features: Why the Atari 2600?21
Preparing for the Hack23
Performing the Hack29
Opening the Case29
Cleaning the Case31
Mocking Up the Design32
Configuring the BIOS35
Installing Software36
Preparing the Control Panel38
Preparing the USB/FireWire Backplane45
Preparing the Cordless Keyboard/Mouse Receiver46
Preparing the Stelladaptor 2600 Controller-to-USB Interfaces	.51
Preparing the Power Supply Connector54
Preparing the Mini-ITX Motherboard56
Preparing the Housing59
Putting It All Together67
The CD-ROM Drive67

The Motherboard	70
The Hard Drive	71
The PW70 Power Supply Module	72
The USB Components	73
The Control Panel	75
Closing It Up: Completing the Atari 2600PC Case Modification	78
In Conclusion...	82
Resources and Other Hacks	82
Case Modifications on the Web	82
Stuffing PCs into Videogame System Consoles	83
Creating Your Own Portable Game System	83
Parts and Materials	83
Part II Modern Game Consoles	85
Chapter 3 The Xbox	87
Introduction	88
Xbox Hardware and Specifications	89
Xbox Versions	90
Opening the Xbox	92
Preparing for the Hack	92
Performing the Hack	92
Controller Hacks	95
Controller Versions	96
Getting Inside Your Controller	97
Preparing for the Hack	97
Performing the Hack	97
Illuminating the Controller Buttons with LEDs	99
Preparing for the Hack	99
Performing the Hack	99
Under the Hood: How the Hack Works	103
Testing and Troubleshooting	104
Optional Hack: Illuminating the Controller Logo	104
Adding a Remote Reset Switch	104
Adding a Remote Reset Switch to the Xbox Controller	104
Preparing for the Hack	104
Performing the Hack	105

Adding a Remote Reset Switch to the Xbox Controller	
Memory Card or Xbox Live Communicator	107
Preparing for the Hack	107
Performing the Hack	108
Testing and Troubleshooting	110
Adding an Xbox Live Communicator to a Wireless	
Controller	111
Preparing for the Hack	111
Performing the Hack	112
Xbox Networking Hacks	112
Establishing a Network Link Using Standard Networking	113
Performing the Hack	114
Testing and Troubleshooting	115
Creating Your Own Crossover Cable	116
Preparing for the Hack	117
Performing the Hack	117
Testing and Troubleshooting	119
Extending the Network Status LEDs to the Front Panel	120
Preparing for the Hack	120
Performing the Hack	120
Testing and Troubleshooting	122
Wireless Networking Hacks	123
Adding a Wireless Networking Adapter to the Xbox	123
Adding a Removable Antenna to the Microsoft Xbox	
Wireless Adapter	125
Preparing for the Hack	126
Performing the Hack	126
Under the Hood: How the Hack Works	131
Installing a Modchip	131
A Brief Introduction to Modchips	131
Preparing for the Hack	135
Performing the Hack	135
Running Linux on an Unmodified Xbox	141
Preparing for the Hack	141
Performing the Hack	142
Other Hacks	144
Homebrew Game Development	144
Xbox Resources on the Web	146

Chapter 4 PlayStation 2	147
Introduction	148
Commercial Hardware Hacking: Modchips	148
Getting Inside the PS2	150
Mainboard Revisions	150
Identifying Your Mainboard	151
Opening the PS2	152
Installing a Serial Port	156
Preparing for the Hack	157
Performing the Hack	158
Testing	164
Under the Hood: How the Hack Works	164
Booting Code from the Memory Card	164
Preparing for the Hack	165
Performing the Hack	165
Preparing TITLE.DB	165
Choosing BOOT.ELF	168
Saving TITLE.DB to the Memory Card	168
Independence!	169
Under the Hood: How the Hack Works	169
Other Hacks: Independent Hard Drives	171
PS2 Technical Details	171
Understanding the Emotion Engine	172
The Serial I/O Port	173
The I/O Processor	175
The Sub-CPU Interface	176
Homebrew Game Development	176
PS2 Resources on the Web	177
Part III Handheld Game Platforms	179
Chapter 5 Nintendo Game Boy Advance	181
Introduction	182
Game Boy, 1989	182
Game Boy Pocket, 1996	183
Game Boy Color, 1998	183
Game Boy Advance, 2001	184
Game Boy Advance SP, 2003	185

A Very Brief History of Nintendo	186
Opening the GBA Console	187
Preparing for the Hack	187
Performing the Hack	188
Replacing the Display Lens	193
Preparing for the Hack	194
Performing the Hack	194
Light Up Your LCD with the GBA Afterburner Mod	198
Preparing for the Hack	198
Performing the Hack	200
Removing the LCD	201
Preparing the GBA Housing	203
Preparing the LCD	206
Preparing the Afterburner Module	209
Installing the Afterburner Module	211
Adding the Brightness Control (Optional)	214
Under the Hood: How the Hack Works	216
Enhancing Your Afterburner with the GBA Stealth Dimmer Chip	217
Preparing for the Hack	218
Performing the Hack	219
Under the Hood: How the Hack Works	225
Nintendo GBA Technical Specifications	226
The Central Processor	226
CPU Registers	227
Memory Architecture	228
Internal Working RAM	229
External Working RAM	230
Graphics Memory	230
Game ROM and Game Save Memory	231
The Graphics System	231
Tile-Based Modes (0–2)	232
Bitmap-Based Modes (3–5)	232
The Sound System	233
Homebrew Game Development	233
Other Hacks	234
Nintendo GBA Resources on the Web	238

Chapter 6 Gamepark 32 (GP32)	241
Introduction	242
Out of the Box: Configuring Your GP32	245
Opening the GP32 Console	251
Preparing for the Hack	251
Performing the Hack	251
Replacing the GP32 Screen Cover	257
Preparing for the Hack	258
Performing the Hack	258
Repairing Your Buttons	262
Preparing for the Hack	262
Performing the Hack	263
Accelerating Your GP32 (CPU Core Voltage Increase)	264
Preparing for the Hack	265
Performing the Hack	265
Under the Hood: How the Hack Works	268
Creating a DC Power Adapter	269
Preparing for the Hack	269
Performing the Hack	273
Under the Hood: How the Hack Works	275
Installing the Multifirmware Loader	275
Preparing for the Hack	276
Performing the Hack	276
Backing Up Your Firmware	276
Reprogramming (Flashing) the New Firmware	278
Homebrew Game Development	280
Other Hacks	284
GP32 Resources on the Web	286
Part IV Retro and Classic Systems	289
Chapter 7 Nintendo NES	291
Introduction	292
Opening the NES Console	294
Preparing for the Hack	294
Performing the Hack	294
Replacing the 72-Pin Cartridge Connector	299
Preparing for the Hack	300

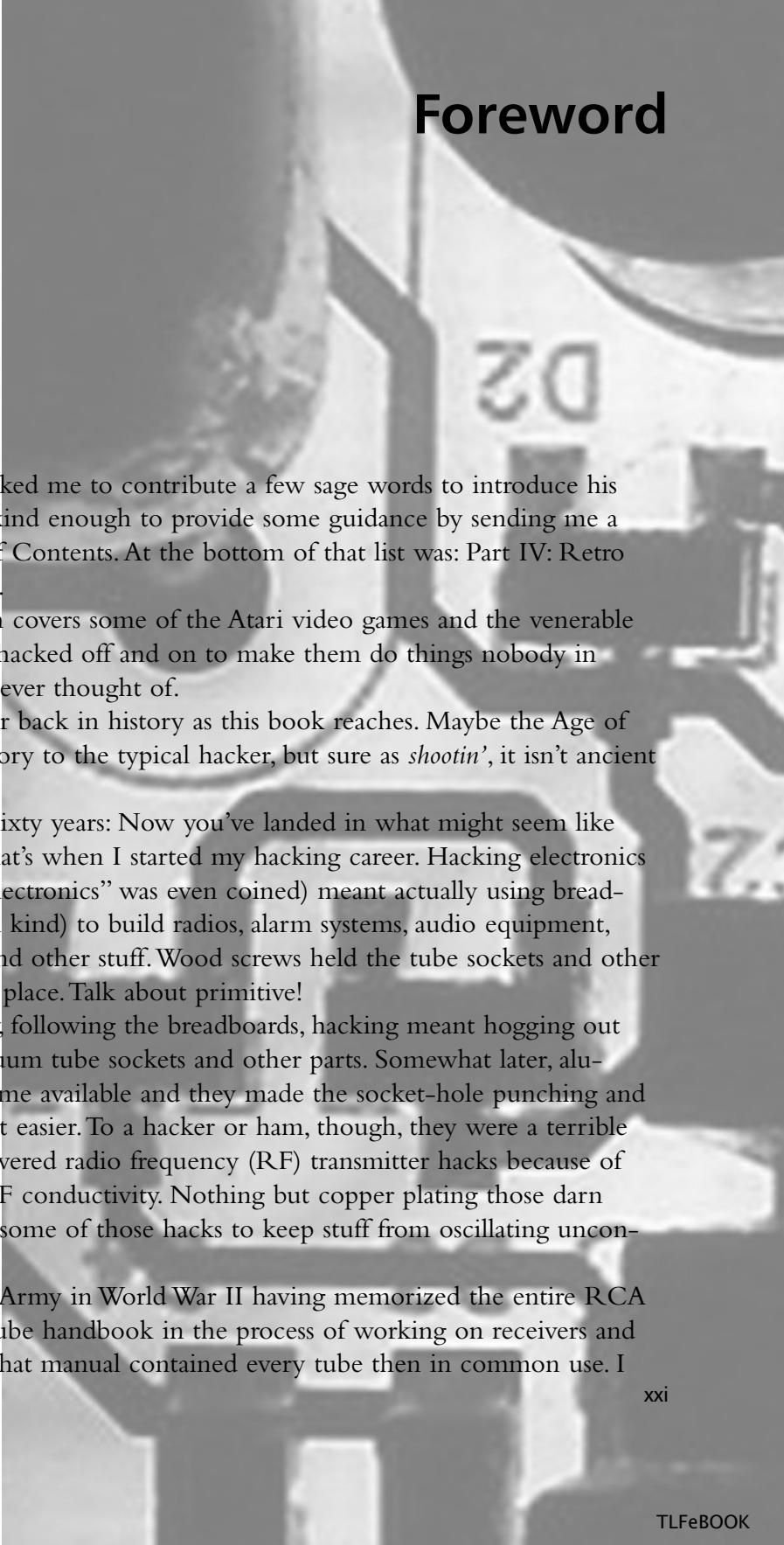
Performing the Hack	301
Blue Power LED Modification	302
Preparing for the Hack	303
Performing the Hack	304
Under the Hood: How the Hack Works	310
Disabling the NES “Lockout Chip”	311
Preparing for the Hack	312
Performing the Hack	312
Optional: Adding a Switch	315
Under the Hood: How the Hack Works	315
Opening an NES Game Cartridge	316
Preparing for the Hack	316
Performing the Hack	318
Replacing the Battery in Certain Game Cartridges	319
Preparing for the Hack	320
Performing the Hack	321
Creating an EPROM Cartridge for Homebrew Game	
Development	324
Preparing for the Hack	324
Performing the Hack	325
Under the Hood: How the Hack Works	330
Homebrew Game Development	330
Other Hacks	332
NES Resources on the Web	333
Chapter 8 Atari 2600	335
Introduction	336
Hacks in This Chapter	337
Atari 2600 Left-Handed Joystick Modification	337
Preparing for the Hack	338
Performing the Hack	338
Repair Your Atari 2600 Joysticks	342
Preparing for the Hack	342
Performing the Hack	343
Revitalize Your Atari 2600 Paddles	349
Preparing for the Hack	350
Performing the Hack	350
Use an NES Control Pad with your 2600	356

Preparing for the Hack	357
Performing the Hack	358
Atari 2600 S-Video/Audio Mod	364
Preparing for the Hack	364
Performing the Hack	366
Optional: Commodore 1702 Hack	380
Optional: Do-It-Yourself 2600 A/V Mod	381
Technical Information	381
Atari 2600 Stereo Audio Output	382
Preparing for the Hack	384
Performing the Hack	384
Under the Hood: How the Hack Works	391
Homebrew Game Development	391
Atari 2600 Resources on the Web	396
Chapter 9 Atari 5200	399
Introduction	400
Opening the Atari 5200	401
Preparing for the Hack	401
Performing the Hack	401
Reassembly	408
Atari 5200 Blue LED Modification	408
Preparing for the Hack	409
Performing the Hack	410
Under the Hood: How the Hack Works	413
Atari 5200 Two-Port BIOS Replacement	413
Preparing for the Hack	414
Performing the Hack	414
Creating an Atari 5200 Paddle Controller	419
Preparing for the Hack	421
Performing the Hack	421
Disassembling the Atari 2600 Paddle Controller	422
Building the 5200 Paddle Controller	424
Adding a Weighted Dial	432
Under the Hood: How the Hack Works	433
Freeing Yourself from the 5200 Four-Port Switchbox	434
Preparing for the Hack	435

Performing the Hack	436
Under the Hood: How the Hack Works	445
Atari 5200 Video and Audio Upgrade Modification	446
Preparing for the Hack	447
Performing the Hack	449
Other Hacks	467
Rebuilding Atari 5200 Controllers	467
Atari 5200 Four-Port VCS Cartridge Adapter Fix	470
Homebrew Game Development	470
Atari Resources on the Web	474
Chapter 10 Atari 7800	477
Introduction	478
Hacks in This Chapter	479
Blue LED Modification	479
Preparing for the Hack	480
Performing the Hack	481
Under the Hood: How the Hack Works	485
Game Compatibility Hack to Play Certain Atari 2600 Games	486
Preparing for the Hack	487
Performing the Hack	487
Under the Hood: How the Hack Works	489
Voltage Regulator Replacement	490
Preparing for the Hack	490
Performing the Hack	491
Under the Hood: How the Hack Works	494
Power Supply Plug Retrofit	495
Preparing for the Hack	496
Performing the Hack	497
Other Hacks	501
Atari 7800 Composite and S-Video Output	501
Sega Genesis to Atari 7800 Controller Modification	501
NES Control Pad to Atari 7800 Controller Modification	502
Atari 7800 DevOS Modification and Cable Creation	502
Homebrew Game Development	502
Atari 7800 Resources on the Web	506

Appendix A Electrical Engineering Basics	.509
Introduction	.510
Fundamentals	.510
Bits, Bytes, and Nibbles	.510
Reading Schematics	.514
Voltage, Current, and Resistance	.516
Direct Current and Alternating Current	.517
Resistance	.518
Ohm's Law	.518
Basic Device Theory	.519
Resistors	.519
Capacitors	.521
Diodes	.524
Transistors	.526
Integrated Circuits	.528
Microprocessors and Embedded Systems	.530
Soldering Techniques	.531
Hands-On Example: Soldering a Resistor to a Circuit Board	.531
Desoldering Tips	.533
Hands-On Example: SMD Removal Using ChipQuik	.534
Common Engineering Mistakes	.537
Web Links and Other Resources	.538
General Electrical Engineering Books	.538
Electrical Engineering Web Sites	.539
Data Sheets and Component Information	.539
Major Electronic Component and Parts Distributors	.540
Obsolete and Hard-to-Find Component Distributors	.540
Appendix B: Coding 101 and Appendix C: Operating Systems Overview are available via the companion website at www.syngress.com/solutions.	
Index	541

Foreword



When Joe Grand asked me to contribute a few sage words to introduce his new book, he was kind enough to provide some guidance by sending me a preliminary Table of Contents. At the bottom of that list was: Part IV: Retro and Classic Systems.

That last section covers some of the Atari video games and the venerable NES which I have hacked off and on to make them do things nobody in California or Japan ever thought of.

Now, that's as far back in history as this book reaches. Maybe the Age of Atari is ancient history to the typical hacker, but sure as *shootin'*, it isn't ancient history to me!

Go back some sixty years: Now you've landed in what might seem like prehistoric times; that's when I started my hacking career. Hacking electronics (before the term "electronics" was even coined) meant actually using breadboards (the wooden kind) to build radios, alarm systems, audio equipment, motor controllers and other stuff. Wood screws held the tube sockets and other mechanical parts in place. Talk about primitive!

Chronologically, following the breadboards, hacking meant hogging out steel chassis for vacuum tube sockets and other parts. Somewhat later, aluminum chassis became available and they made the socket-hole punching and parts mounting a lot easier. To a hacker or ham, though, they were a terrible choice for high-powered radio frequency (RF) transmitter hacks because of aluminum's poor RF conductivity. Nothing but copper plating those darn chassis would tame some of those hacks to keep stuff from oscillating uncontrollably.

I went into the Army in World War II having memorized the entire RCA receiving vacuum tube handbook in the process of working on receivers and audio equipment. That manual contained every tube then in common use. I

knew the whole book inside out. Try that nowadays with a list of discrete components, ICs and micros. It's scary how far we have come.

Talking about scary experiences while hacking:

Back in the late thirties I built an RF oscillator-AM modulator and fed the crystal pick-up of my 78 RPM phonograph turntable into it. My test record was a 10 inch (not 12 inch) shellac record of the Andrew Sisters singing the "Beer Barrel Polka." The first time I tested that gadget, it worked like a charm, playing the music through my crappy little 4-tube radio. Then the unexpected happened; as soon as the song was finished and I shut down the power on my hack, the Beer Barrel Polka started playing all over again. That made the short hairs on my neck stand up for a few seconds. Had those radio waves been bouncing around my room and come back to life? Then I figured it out; I had been suppressing a local radio station with my transmission. When I shut down my RF oscillator, a radio station came on and, quite coincidentally, started up that same, then ever-so-popular recording.

Vacuum tubes gave way to transistors in the fifties and I had to shift gears. The first piece of hardware I hacked in the early fifties used point-contact transistors. The doggone circuit took off and started working before I could even hook up a power supply. There was so much RF from nearby TV and radio transmitters floating around downtown New York that the long wires of the hack, acting as antennas, picked up enough energy for the transistors to self-rectify it and powered up the circuitry. Now don't think that didn't give me the willies until I figured out what was going on!

We no sooner got the hang of transistors when the first generation of ICs came along. Some worked, some didn't...it took a few years to get that straightened out. We went from RC-coupled ICs from TI to DTL made by Fairchild to TTL by Sylvania and occasionally had to use ECL logic from Motorola when high speed (10 MHz or so....ha!) was needed.

That was in the fifties and the sixties. Microprocessors had not been born yet. Everything we built then was in hardware. Software? What was that? Something some guys screwed around with at universities and in big companies where one of those refrigerator size mainframe monsters was available for research purposes.

It was during this transition period that home video games were born.

Actually, the thought of doing something interactive with a TV set had dawned on me much earlier. I was hired to design and build a TV set at Loral

back in the early fifties, working with another engineer. I thought we could distinguish our set from the rest of them by doing something novel, like moving a couple of spots around on the screen to play a car racing game or whatever. Management's reaction was predictable: "Forget it. Finish the damn set. You're behind schedule as it is."

The thought resurfaced in August of 1966. I wrote a 4-page disclosure document on September 1st that laid it all out: Chase games, sports games, quasi-board games...the lot! I had one of the engineers in my division at Sanders Associates sign and date each page. That document started a whole new industry...but who knew that at the time.

For me, that was going to be the hack to keep me from going nuts. I was running a division with some five hundred engineers, techs and support people. We were busy cranking out designs for defense electronics such as radar, electronic counter-measure and anti-submarine warfare equipment.

My opportunity to get close to the bench and actually work on something hands-on was vanishingly close to zero. What to do to keep from getting stale? Hack something, of course.

Now, being the manager of a large operation has some advantages. You can do a certain amount of skunk work without rippling the overhead significantly...so that's what I did.

To those of you who are accustomed to hacking into today's fancy gear, what followed next must seem like a complete anachronism. I put a tech on a bench in a small lab, gave him a key to the door and told him to build some delay-multivibrator (MV) circuitry, drive it with vertical and horizontal sync pulses from a Heathkit TV set alignment generator, sum the MV outputs into the modulator of the Heathkit and see if we could move a spot around the screen. He did what I asked him to do and it worked. I had him use four dual triodes to display a spot on the screen and move it around with H and V control; and to add some color to the spot or to the background - the basics of video game action. Why vacuum tubes and not transistors? Because that alignment generator was a vacuum tube device and also because I still had one foot in the tube age.

After we had a spot, which we could move around the screen and could be colored at will, our preliminary learning experience was over. Now the question was: What do we build that might actually become a real product, a TV Game?

Little did I know then that this clandestine hack was the start of a three-year trip, mostly part-time, that would finally take the form of a switch-programmable piece of hardware capable of delivering Ping-Pong, Handball, Volleyball, Chase and Gun games. We called that the Brown Box because we had covered it with self-adhesive, brown wood grain paper to make it look halfway presentable. That venerable Brown Box now lives on at the Smithsonian among other relics of the birth of video games.

Now, we were at a stage where management had to get clued in. You can't hide things forever. In early '67, our first go-around with chase games and gun games was ready for show-and-tell. Being a true hacker I couldn't resist adding a 4.5 MHz FM oscillator to our chassis. It was already packed full of discrete transistor circuitry, but we found a place to squeeze in another small board. This FM'ed RF oscillator was applied as another modulating signal to the Channel 3 oscillator of our game. The FM oscillator was in turn driven by the output of a tape recorder. That allowed me to make a tape recording on which I introduced each of the games in my best announcer's voice. Applying the 4.5 MHz FM oscillator's output to the Channel 3 RF oscillator creates RF carrier components 4.5 MHz above and below the video signal carrier frequency. One of these is in the right spectrum to get through a TV set and gets treated like a legitimate sound signal. So here we had the first home video game presentation anywhere, ever...and it had voice-over game announcements coming through the TV set's loudspeaker. Neat!

It happened that the Board of Directors was meeting the day we were scheduled to present this game system to the President and the Executive V.P. for whom I worked at the time. He was none too happy to see me screw around with this stuff that had nothing to do with the real work at Sanders Associates. When the demonstration began, we had an unexpected audience of a dozen people: The entire Board was there as were some hangers-on. I was doubly glad I had hacked the voice-over scheme so I wouldn't bungle the presentation.

The reaction was what you might expect: A lot of raised eyebrows and the enthusiastic support of at least one member of the Board who thought that it was about time that Sanders Associates did something *out of the box*. Well, it sure was.

Now, hacking is one thing. Making a product for sale on the open market or licensing it to someone who will do it for you, that's quite another thing.

It took three long years to find a licensee who would go forward and spend the million bucks required to do market testing, production engineering, tooling,

distribution and marketing; and that was Magnavox. The first Magnavox Odyssey games showed up in stores in the fall of 1972, over five years after I had the original epiphany. A couple of years and about 340,000 games later, Odyssey was replaced by a newer model using IC's and the competition was busy cranking out their own versions. The industry had been launched. The fact that Atari's Pong arcade game hit the street in 1973 and caused the arcade video game business to take off like a big bird, that didn't hurt Odyssey sales one bit.

Any hacker who has ever looked into the Magnavox Odyssey game had to ask him or herself: "How did this thing ever get into production in 1972?" Why wasn't it full of CMOS instead of discrete components? There are some 40 transistors for the flip-flops and one-shots needed to generate the sync signals as well as the player-controlled and the machine-controlled screen symbols, and some 40 diodes connected in different ways by plug-in game cards that changed the logic of the circuitry to produce the desired game action.

Well, it's simple. Our design was of 1967 vintage; we were done in early 1968 but could not find a taker until 1969 when we demonstrated it to every U.S. TV set maker and eventually got into bed with Magnavox. Then another year was spent with the lawyers dickering about who struck John and now we're into 1971. Finally, extensive field-testing for consumer acceptance of this unknown category of product chewed up another half year. The response was very positive. So then a small group of engineers culled from the Fort Wayne TV set design department were given the job to redesign our Brown Box for production. They were told to get this thing into production by early '72. Now they were down to a few months to get the job done. They did what any sensible hacker would have done. They copied the Brown Box almost part-for-part and made changes only to increase stability and meet some FCC specs that applied to the novel product.

That's how an ancient transistor design survived for nine years and was almost an antique before production was halted in the spring of '75. No one in his right mind would have hacked a design like that in the age of cheap ICs, never mind the first generation single-chip state machines that were becoming cost-effective.

Well, it *was* an ancient design but it worked. The plug-in card method of interconnecting the internal logic allowed some creative hackers to come up with additional games that were not sold with the first lot of Odysseys. I sat down in my own lab during the winter of '72 to '73 and hacked two new plug-in cards that made use of the novel idea of putting "active" circuitry on the card—not just novel interconnections. One of those cards was an improvement over the basic

Ping-Pong card. My new circuitry took the signal off the ball direction-reversing flip-flop and used it to twang a “pong” sound (Atari, please excuse the expression). I mounted the required electronic circuitry and a tiny speaker on the back of the “active” plug-in card. While I was at it, I also reached into the speed control circuit of the ball spot and added two pots with which the players could tweak their ball speed individually. Then I demonstrated the card to Magnavox. It drew a big yawn. So did a second “active” card which allowed the basic handball game to produce ball-slapping sounds and added a feature which caused the wall to gradually move closer to the players, speeding the game up progressively. It was fun to play, also and drew the same amount of enthusiasm from the great *marketeers* at Magnavox.

You can take a horse to water but you can't make it drink, I guess.

Fortunately the TV game engineers at Magnavox, now labeled video game engineers, were true hackers and were ready with next-generation IC designs before management even stopped dithering on whether they wanted to be in this business for the long haul.

Comparing those early game systems with a PS2 or an Xbox is in the same league as comparing a Model T with the Mars Rover. It's definitely a mite harder nowadays to get your arms around a modern video game system and hack it, but that won't stop us.

With best wishes to all hackers everywhere.

— Ralph H. Baer
The Father of Video Games
<www.ralphbaer.com>

Introduction 2.0

The way we customize our things says a lot about who we are.

Today, everywhere we look, we are surrounded by a convergence of media – videogames, advertisements, and television. We are told what to believe, how to think, and how to act. We are told what's cool and what's not, what we should buy, what we should wear, and what music we should listen to.

Hardware hacking has never been about what the mainstream media thinks. It's about creativity, education, experimentation, personalization, and just having fun. This book is no different.

Game Console Hacking focuses on modifying our favorite videogame systems to do things they were never intended to do, to add features that we've always wanted but the vendors never gave us, or to create something that has never been done before.

This book is a little bit different than what you might be used to. We cover a wide spectrum of gaming consoles, from the retro and arguably archaic Atari systems, to the teenaged Nintendo NES console, up through the modern consoles like Xbox and PlayStation 2. There's something in here for every type of gamer, whether you like to get your hands dirty with modifying hardware or whether you're an aspiring game developer. Step-by-step hacks are presented with a slew of pictures to hold your hand along the way, as well as resources to let you jump right in to creating your own games for the systems. It's all about education and inspiring you, the reader, to break the mold of what's considered "acceptable." And best of all, you can do so in the comfort of your own home, without breaking any laws.

Long gone are the days where a few guys can make millions on a self-published videogame they designed in Mom's garage. But, the thrill for homebrew game development is still there; and, it has close ties to hardware hacking in that you are giving the system a touch of your personal creativity, doing things the way you want to. It gives us a sense of ownership that a faceless company can't provide.

There is an underbelly to the videogame industry, which nowadays just seems to only sell multi-million dollar productions with gameplay based on franchise licenses and the same, overused 3D game engines. There are thriving development communities for all the systems we cover in this book. There are people who still yearn to develop games just so they can *play* those games. Sharing code samples, socializing with fellow programmers, hacking videogame systems to allow them to run their custom software, designing games for the sheer thrill of the kill. For gamers, by gamers.

There's something to be said for pouring your heart and soul into a creative game design or hardware hack, and I hope this book will entice you to do so. Inspiration and creativity can't be taught or forced. The possibilities are endless.

The way we customize our things says a lot about who we are.

Who are you?

—Joe Grand, author, hardware hacker, and gamer
July 2004

Introduction 1.0

Hardware hacking. Mods. Tweaks. Though the terminology is new, the concepts are not: A gearhead in the 1950s adding a custom paint job and turbo-charged engine to his Chevy Fleetline, a '70s teen converting his ordinary bedroom into a “disco palace of love,” complete with strobe lights and a high-fidelity eight-track system, or a techno-geek today customizing his computer case to add fluorescent lighting and slick artwork. Taking an ordinary piece of equipment and turning it into a personal work of art. Building on an existing idea to create something better. These types of self-expression can be found throughout recorded history.

When Syngress approached me to write *Hardware Hacking: Have Fun While Voiding Your Warranty*, our first book on hardware hacking, I knew they had hit the nail on the head. Where else could a geek like me become an artistic genius? Combining technology with creativity and a little bit of skill opened up the doors to a whole new world: hardware hacking.

But why do we do it? The reasons might be different for all of us, but the result is usually the same. We end up with a unique thing that we can call our own—imagined in our minds and crafted through hours, days, or years of effort. *And* doing it on our own terms.

Hardware hacking today has hit the mainstream market like never before. Computer stores sell accessories to customize your desktop PC. Web sites are popping up like unemployed stock brokers to show off the latest hacks. Just about any piece of hardware can serve as a candidate to be hacked. Creativity and determination can get you much farther than most product developers could ever imagine. Hardware hacking is usually an individual effort, like creating a piece

of art. However, just like artists, hackers sometimes collaborate and form communities of folks working toward a similar goal.

The use of the term *hacker* is a double-edged sword and often carries a mythical feel. Contrary to the way major media outlets enjoy using the word to describe criminals breaking into computer systems, a hacker can simply be defined as somebody involved in the exploration of technology. And a *hack* in the technology world usually defines a new and novel creation or method of solving a problem, typically in an unorthodox fashion.

The philosophy of most hardware hackers is straightforward:

- Do something with a piece of hardware that has never been done before.
- Create something extraordinary.
- Harm nobody in the process.

Hardware hacking arguably dates back almost 200 years. Charles Babbage created his difference engine in the early 1800s—a mechanical form of hardware hacking. William Crookes discovered the electron in the mid-1800s—possibly the first form of electronics-related hardware hacking. Throughout the development of wireless telegraphy, vacuum tubes, radio, television, and transistors, there have been hardware hackers—Benjamin Franklin, Thomas Edison, and Alexander Graham Bell, to name a few. As the newest computers of the mid-20th century were developed, the ENIAC, UNIVAC, and IBM mainframes, people from those academic institutions fortunate enough to have the hardware came out in droves to experiment. With the development and release of the first microprocessor (Intel 4004) in November 1971, the general public finally got a taste of computing. The potential for hardware hacking has grown tremendously in the past decade as computers and technology have become more intertwined with the mainstream and everyday living.

Hardware hacks can be classified into four different categories, though sometimes a hack falls into more than one:

1. **Personalization and customization** Think “hot rodding for geeks,” the most prevalent of hardware hacking. This includes things such as case modifications, custom skins and ring tones, and art projects like creating an aquarium out of a vintage computer.
2. **Adding functionality** Making the system or product do something it wasn’t intended to do. This includes things such as converting the iPod to run Linux, implementing a serial port interface on your PlayStation 2, or modifying the Atari 2600 to support stereo sound.
3. **Capacity or performance increase** Enhancing or otherwise upgrading a product. This includes things such as adding memory to your favorite personal dig-

ital assistant (PDA), modifying your wireless network card to support an external antenna, or overclocking your PC's motherboard.

4. **Defeating protection and security mechanisms** This includes things such as removing the unique identifier from CueCat barcode scanners, finding Easter eggs and hidden menus in a TiVo or DVD player, or creating a custom cable to unlock the secrets of your cell phone.

Creating your own hardware hacks and product modifications requires at least a basic knowledge of hacking techniques, reverse engineering skills, and a background in electronics and coding. All the information you'll need is in the pages of this book. And if a topic isn't covered in intimate detail, we include references to materials that do. If you just want to do the hack without worrying about the underlying theory behind it, you can do that, too. The step-by-step sections throughout each chapter include pictures and "how to" instructions. The details are in separate sections that you can skip right over and get to the fun part—voiding your warranty!

This book has something for everyone from the beginner hobbyist with little to no electronics or coding experience to the self-proclaimed "gadget geek" and advanced technologist. It is one of the first books to bring hardware hacking to the mainstream. It is meant to be fun and will demystify many of the hacks you have seen and heard about. We, all the contributors to this project, hope you enjoy reading this book and that you find the hacks as exciting and satisfying as we have.

If your friends say "Damn, now *that's cool*," then you know you've done it right.

—Joe Grand, the hardware hacker formerly known as Kingpin
January 2004

Introduction to Hardware Hacking



Tools of the Warranty-Voiding Trade

Topics in this Chapter:

- Introduction
- The Essential Tools
- Basic Hardware Hacking
- Advanced Projects and Reverse Engineering
- Where to Obtain the Tools

Introduction

Before you start your game console hacking projects, you'll need the right arsenal of tools. For some hacks, you might need only a single screwdriver. For others, you could need a workshop complete with power tools and advanced electronic equipment. For the most part, it isn't necessary to have a world-class laboratory or top-of-the-line computer system to conduct most levels of game console hacking. However, it's amazing how much easier things are if you have the right tools for the job.

Besides the physical tools you will need for hardware hacking that we list in this chapter, you'll need a computer system for any adventures into homebrew game development. After deciding on the game console you'll be programming for, you can choose your development system based on the tools that you'll need. Depending on the console you are writing games for, the appropriate development tools might run only on a specific platform (such as Windows, Macintosh, or Linux). Typically, a desktop or laptop PC running Windows 2000/XP with minimum specifications of 1GHz processor, 256MB RAM, 20GB hard drive, and decent graphics card will be sufficient. The more complex and processor-intensive the development tool or emulator, the more powerful your machine will need to be.

The tools and supplies listed in this chapter are merely a baseline of any good hardware hacking cache. We don't list every possible tool in existence, because there is usually more than one solution to any given problem. Think of this section as telling you about the supplies you'll want in your "kitchen," with each hack containing the actual "recipe" you'll cook with. Each hack presented in this book provides a list of the specific tools and components you'll need to pull it off.

We include a selection of pictures that show some of the more unique tools of the warranty-voiding trade. These lists will give you an idea of what you'll need to get a good start so you can jump in and get down to hacking.

We have separated the listings into three parts:

- The Essential Tools
- Basic Hardware Hacking
- Advanced Projects and Reverse Engineering

The work area where your activities take place should be a clean, smooth, and well-lit area where you can easily organize and handle parts and/or documentation without losing them. An inexpensive sheet of white poster board makes an excellent construction surface while providing protection for the underlying table or desk.

WARNING: PERSONAL INJURY



Safety is an important consideration. With many of the tools listed here, improper or careless use can lead to accidents and personal injury. Please take the time to read all necessary instruction manuals and safety documentation before starting your hack. Be sure to wear protective gear at all times, keep your work area free of unnecessary clutter, use a suitable stand for your soldering iron, and avoid tangling the cords of your various tools.

The Essential Tools

The following are some essential tools for the beginner hardware hacker—someone who is curious about dabbling in and experimenting with simple hacks. It always helps to have a good stock of various equipment, wires, tools, components, and other materials in your workshop so you don't have to run out to the store every time you need something. Here are the basics:

- **Bright overhead lighting or desk lamp** Well-diffused overhead lighting is recommended—bright white fluorescent or incandescent bulbs serve this purpose. A smaller, high-intensity desk lamp will prove especially helpful for close-up work.
- **Protective gear** Mask or respirator, goggles, rubber gloves, smock or lab coat, earplugs. A sampling of protective gear is shown in Figure 1.1. Such gear should be worn at all times when performing your hacks. Use the respirator to prevent breathing in noxious fumes and fine dust from painting, cleaning, cutting, or soldering. The goggles protect your eyes from stray plastic or wood chips during drilling. Use the smock to prevent damage (burns and stains) to clothing.

Figure 1.1 Protective Gear



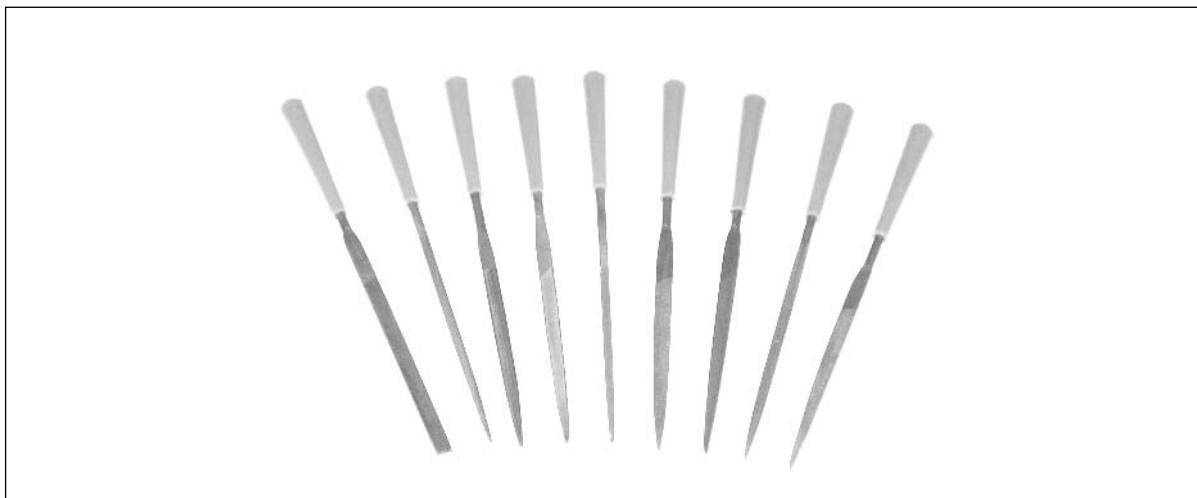
- **Electrostatic discharge (ESD) protection** If you live in a dry environment that is prone to static electricity, it is recommended that you purchase an antistatic mat and wrist strap from a local electronics store to prevent static discharge and protect sensitive electronic circuitry from getting damaged. Make sure the antistatic mat is properly grounded so that it can serve its intended purpose. Think of walking on a shag rug in your bare feet and then touching the radiator or a sibling. You'll feel ESD at work. However, ESD can damage components, even if you don't feel anything. You don't want that happening to the device you're hacking.

- **Screwdrivers** Regular-sized Phillips and flat head screwdrivers and a smaller set of jeweler's screwdrivers. The more sizes and types, the better, because you never know what sorts of hardware you'll want to open.
- **X-ACTO hobby knife** The modeling tool of choice for crafters, artists, and hobbyists. An essential general-purpose tool, especially useful for case mods and circuit board hacks. Over 50 different blade types are available.
- **Dremel tool** Extremely useful carving tool. Helpful for case mods and opening housings. Some models support rotation speeds from single-digit revolutions per second up to tens of thousands. Many various bit types (drilling, sanding, carving, engraving), accessories, and attachments are available. Example: Dremel 395 Variable-Speed MultiPro, \$74.99 (see Figure 1.2).

Figure 1.2 Dremel Tool



- **Needle file set** Designed for precise filing (see Figure 1.3). Ideal for deburring drilled holes and preparing modified surfaces. Most five-piece sets include square, flat, triangle, round, and elliptical files. Example: Radio Shack Kronus 5-Piece Needle File Set #64-2977, \$7.99.
- **Tweezers** Handy for dealing with small components, holding wires, and pulling out splinters. There are dozens of tweezer styles, including long, extra long, flat tipped, curved, blunt, bent angle, medical, and surgical. The more variety you have in your toolkit, the better.

Figure 1.3 Needle File Set

- **Wire brushes** Great for cleaning tough surfaces, especially metal. Useful for removing rust, dirt, and debris or preparing surfaces to be painted. It is recommended that you have a hand-sized brush for large areas and a smaller toothbrush-shaped brush for more detailed work.
- **Sandpaper** All-purpose sanding sheets are useful for removing dirt and debris, deburring edges, or preparing surfaces to be painted or glued together. An assortment of various grits (for example, 100, 220, 400, and 600) is recommended.
- **Glues** Wood glue, Gorilla Glue, Super Glue, epoxy, hot glue, acrylic cement. The more types of adhesive that you have on hand, the better off you'll be, because some glues work better on certain surfaces than others. A sampling of glues is shown in Figure 1.4.

Figure 1.4 Types of Glue

- **Tape** Duct tape, masking tape, electrical tape, Scotch/transparent tape, double-sided foam tape.
- **Cleaning supplies** A good workspace is a clean workspace. Typical cleaning supplies include cotton swabs, alcohol pads, paper towels, and some type of spray cleaning solution (for example, Fantastik).
- **Miscellaneous mechanical pieces** These are the standard hardware pieces that you'd find in any household workshop: nails, screws, stand-offs/spacers, washers, nuts, and bolts.

Basic Hardware Hacking

The following mid-range tools are what you'll need for more serious hardware hacking.

- **Variable-speed cordless drill** This is the essential multipurpose tool. It's especially useful for case mods. Example: Skil 18V Cordless Drill/Driver #2867 with 3/8-inch keyless chuck and six torque settings, \$69.99 (see Figure 1.5).

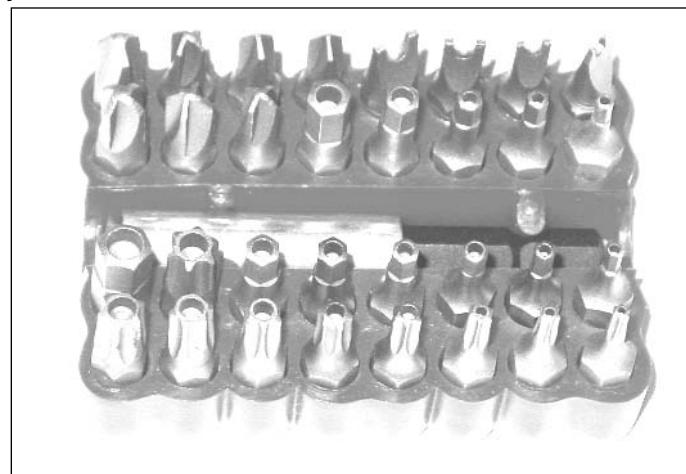
Figure 1.5 Variable-Speed Cordless Drill



- **Drill bit set** What good is your variable-speed cordless drill without a complete set of drill bits of various sizes? Standard sizes include 1/16, 5/64, 3/32, 7/64, 1/8, 9/64, 5/32, 11/64, 3/16, 1/4, 7/32, 5/16, and 3/8 inch. Example: Black & Decker General Purpose 17-Piece Drill Bit Set, \$18.95.

- **Security driver bit set** Security and tamper-resistant screws are sometimes used on product housings to prevent them from being easily opened. There are many types of these specially shaped bits (see Figure 1.6). To identify a particular bit type you might need to use for a hack, visit www.lara.com/reviews/screwtypes.htm.

Figure 1.6 Security Driver Bit Set



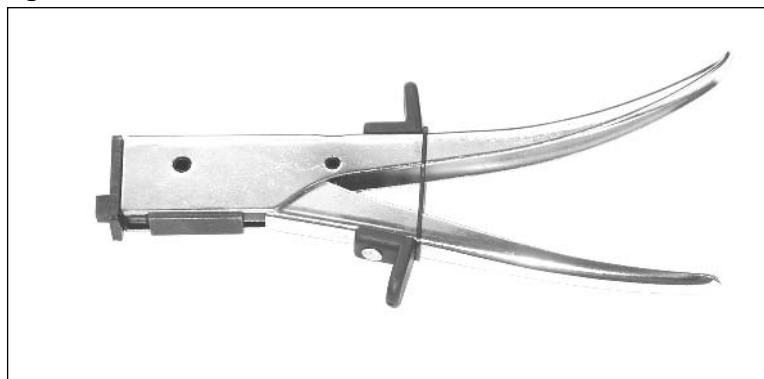
- **Heat gun and heat-shrink tubing** Heat guns look a lot like hair dryers, but, as many instructions thoughtfully point out, they should never be used for drying hair. Heat guns provide an extremely hot, directed flow of air through a nozzle (see Figure 1.7). They are commonly used for removing paint, melting glue, quickly drying surfaces, and shrinking heat-shrink tubing and plastic film. Basic heat guns have single temperature and airflow settings. More advanced models have multiple settings, giving you more control based on your intended application. Example: Milwaukee Dual Temperature Heat Gun (570 and 1000 degrees F), \$69.95.

Figure 1.7 Heat Gun



- **Center punch** Used to mark the target drill spot on a drilling surface, which will prevent the drill bit from slipping. Manual or automatic types exist. You could also use a permanent marker, but that won't stop your drill from slipping.
- **Nibbling tool** This tool "nibbles" away at light-gauge sheet metal, copper, aluminum, or plastic with each squeeze of the handle. Good for housing modifications and creating custom shapes. Example: Radio Shack Kronus Nibbling Tool #64-2960, \$12.99 (see Figure 1.8).

Figure 1.8 Nibbling Tool



- **Jigsaw** Essential power tool for cutting and shaping. Useful for large pieces of material for which a smaller saw or drill isn't suitable. Example: Bosch 1587AVSK Top-Handle Jigsaw, \$134.99.
- **Wire strippers** For cutting or stripping 10- to 22-AWG wire. Example: Radio Shack Kronus Gauged Wire Stripper #64-2980, \$7.99 (see Figure 1.9).
- **Wire clippers** Example: Radio Shack Kronus 4.5-inch Mini Diagonal Cutters #64-2951, \$4.99, or Radio Shack Kronus 5-inch Nippy Cutter #64-2959, \$4.99 (see Figure 1.9).
- **Needle-nose pliers** Example: Radio Shack Kronus 6-inch Long-Nose Pliers #64-2954, \$5.99 (see Figure 1.9).

Figure 1.9 Wire Strippers, Clippers, and Pliers

- **Soldering station** Soldering tools, ranging from a simple stick iron to a full-fledged rework station, come in many shapes and sizes (see Figure 1.10). More advanced models include adjustable temperature control, automatic shut-off, and interchangeable tips for various component package types and soldering needs. Recommended is a fine-tip, 700 degree F, 50W soldering stick iron. Approximate price range \$10.00 to \$1,000.00. Example: Weller W60P Controlled-Output Soldering Iron, \$67.95.

Figure 1.10 Soldering Station

- **Soldering accessories** Essential soldering gear includes solder, no-clean flux, desoldering braid, vacuum desoldering tool (a.k.a. “solder sucker”), IC extraction tool, and ChipQuik

SMD removal kit. Solder should be thin gauge (0.032-inch or 0.025-inch diameter) 60/40 rosin core. The no-clean flux is used to provide good heat transfer between the iron and surfaces to be soldered. Flux often helps prevent cold solder joints, a common soldering problem. The desoldering tool is a manual vacuum device that pulls up hot solder, useful for removing components from circuit boards (Radio Shack #64-2098, \$7.29). The IC extraction tool helps lift integrated circuits from the board during removal/desoldering (Radio Shack #276-1581, \$8.39). The ChipQuik kit allows you to remove surface-mount components quickly and easily. Some soldering accessories are shown in Figure 1.11.

Figure 1.11 Soldering Accessories



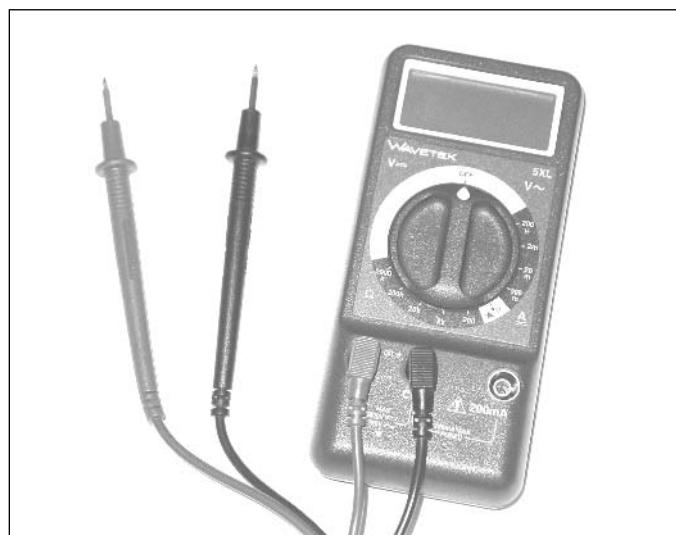
- **Basic electronic components** These include resistors, capacitors, diodes, transistors, light-emitting diodes (LEDs), and switches. It is useful to have a “junk bin” for all sorts of electronics bits and pieces. Old computer equipment and circuit boards are also useful because you can scavenge parts from them as needed. At a minimum, you should have a basic assortment of the most common values of components. Example: Digi-Key 1/4 Watt Resistor Assortment #RS125-ND, \$14.95, and Digi-Key Miniature Electrolytic Capacitor Assortment #P835-KIT-ND, \$29.95.
- **Miscellaneous wires and cables** This category includes cabling and wiring such as test leads, alligator clips, computer cables (USB, serial, parallel), and spools of wire (various colors and lengths, solid or stranded, 20–24AWG).

Advanced Projects and Reverse Engineering

The following tools are for the hardcore hardware hacker who is seriously dedicated to his or her trade. This equipment is mostly targeted toward reverse engineering of circuitry and for use in advanced electronic projects in which you might need to analyze part of a system or create your own circuits. More specific tools exist as well, but generally the tools in this section will get you as far as you need to go for a successful hardware hack of almost any type.

- **Digital multimeter (DMM)** Commonly referred to as the “Swiss army knife” of electronics measurement tools (see Figure 1.12), these are (usually) portable devices that provide a number of precision measurement functions, including AC/DC voltage, resistance, capacitance, current, and continuity. More advanced models also include frequency counters, graphical displays, and digital oscilloscope functionality. Reliable meters have high DC input resistance (also called *input impedance*) of at least 10Mohm. Approximate price range, \$20.00 to \$500.00. Example: Fluke Model 111, \$129.00.

Figure 1.12 Digital Multimeter



- **Analog multimeter** The older siblings to the DMM, these devices provide measurements of AC/DC voltage, resistance, current, and continuity on an analog meter display. Useful for showing slow variations or unusual wave shapes that a DMM may not be able to detect or recognize. Example: Radio Shack Analog Display Compact 8-Range Multimeter #22-218A, \$15.49.
- **Adjustable power supply** Useful for any electronics-related design or hacking. Adjustable, linear, current-limited DC supply (see Figure 1.13). Current limiting often prevents parts from failing (burning up or exploding) when there is a short circuit.

Approximate price range, \$100.00 to \$1,000.00. Example: HP/Agilent Triple Output DC Power Supply E3630A, \$588.00.

Figure 1.13 Adjustable Power Supply



- **Device programmer** Used to read and write memories (RAM, ROM, EPROM, EEPROM, Flash), microcontrollers, and programmable logic devices (see Figure 1.14). Extremely useful to extract program code and stored data. Approximate price range, \$10.00 (home-built) to \$2,500.00. Example: EE Tools' ChipMax, \$345.00.

Figure 1.14 Device Programmer



- **UV EPROM eraser** This tool is used to erase UV-erasable EPROM devices in a matter of minutes using high-intensity ultraviolet light (see Figure 1.15). Approximate price range, \$25.00 to \$250.00. Example: Logical Devices Palm Erase, \$59.95.

Figure 1.15 UV EPROM Eraser



- **PCB etching kit** These kits are used to create printed circuit boards for custom hardware hacks. This process is time consuming and uses hazardous chemicals. Radio Shack provides a kit that contains two 3-inch by 4.5-inch copper-clad circuit boards, resist-ink pen, etching and stripping solutions, etching tank, 1/16-inch drill bit, polishing pad, and complete instructions. PCB etching materials can also be purchased separately at most any electronics distributor. Example: Radio Shack PC Board Kit #276-1576, \$15.49.
- **Oscilloscope** Arguably the most important of advanced measurement tools, this provides a visual display of electrical signals and how they change over time (see Figure 1.16). Available in analog, digital, and mixed-mode versions. Previously owned analog oscilloscopes are typically the most economical and are available at many surplus electronics stores. Look for a bandwidth of greater than 50MHz. Approximate price range, \$100.00 (used) to \$10,000.00. Example: Tektronix 475A 250MHz Analog, \$250.00, or Tektronix TDS3034B 4-Channel 300MHz Color Digital Storage, \$6795.00.

Figure 1.16 Oscilloscope

- **Logic analyzer** An advanced measurement tool useful for concurrently capturing large quantities of digital data from multiple sources. Primarily used for debugging address and data bus access and complex digital circuits. A logic analyzer is characterized by the number of digital samples it can sample at once, the maximum sampling rate, and the maximum sampling depth. Other features include glitch detection, programmable trigger algorithms, and protocol decoding/analysis. Newer systems typically use Windows CE or Windows XP Embedded. Previously owned logic analyzers are the most economical and suitable for most any development or hardware hacking lab—even the “low-end” models serve as excellent diagnostic tools. Approximate price range, \$1000.00 (used) to \$50,000. Example: Hewlett-Packard 1661A, \$1695.00 (used; see Figure 1.17).

Where to Obtain the Tools

This short list of manufacturers and distributors will get you started in finding the supplies you need. The hacks in this book list more specific outlets for each particular type of hardware hack. Your local hardware store, art supply store, hobby shop, or electronic surplus store could also have some useful equipment for you.

Figure 1.17 Logic Analyzer

- The Home Depot, well-known nationwide hardware and home-remodeling chain, www.homedepot.com
- Lowe's, another nationwide hardware and home improvement chain, www.lowes.com
- Hobby Lobby, the nation's largest and most complete creative center; over 60,000 items of arts and crafts supplies, www.hobbylobby.com
- McMaster-Carr, the leading supplier of all things mechanical, including nuts, bolts, washers, lighting, fasteners, hand tools, and raw materials such as metal, ceramic, rubber, plastic, felt, and glass; over 400,000 products to choose from, and 98 percent of those are in stock, www.mcmaster.com
- Radio Shack, well-known supplier of electronic tools, components, and various consumer electronics, www.radioshack.com
- Digi-Key, major distributor for thousands of electronic components, www.digikey.com
- Contact East, leading product distributor for engineering tools, equipment, and materials, www.contacteast.com
- Test Equity, specializing in the sale and rental of used electronic test/measurement equipment, www.testequity.com

Chapter 2

Case Modifications: Building an Atari 2600PC

Topics in this Chapter:

- Introduction
- Choosing Your Features: Why the Atari 2600?
- Building and Atari 2600PC
- Resources and Other Hacks

Introduction

When most people think of hardware hacking, case modifications come to mind. Fans of custom computer case modifications (more easily called *case mods*) have become a huge community in the past few years. In fact, the community, once a small underground group of artistically inclined hackers, has become so large that there are numerous mail-order outlets from which to buy case mod supplies.

Case modding is the ultimate in hardware personalization and expression. Just as many people consider tattoos or piercing to be body art, case mods can most definitely be considered computer art. The canvas may vary, but it is art just the same. You can do a variety of categories of mods, such as:

- **Painting** Adding a custom paint job to your computer case or housing
- **Case windows** Creating “windows” using Plexiglas and edge molding
- **Case and drive lighting** Custom light designs using light-emitting diodes (LEDs) and cold cathode lights
- **Power supply** Modifications of the power supply unit to add a shutdown timer, to change fans, or to add some artistic features
- **Cable management** Adding decorative braided sleeves or rerouting cables in some custom or unique fashion
- **Airflow management** Modifying the case and internal components to allow for increased (or decreased) airflow through the system using fans, fan grills, and fan filters of various sizes
- **Case silencing** Reducing system noise using noise-reduction padding or fan speed controllers

This chapter presents a crash course in case modifications. The goal of the featured hack in this chapter is to cram a fully-featured PC system into a retro Atari 2600 videogame case. Not only is this a real challenge, but it’s extremely rewarding, and you’ll learn a lot about handcrafted case mods in the process!

NEED TO KNOW ... BEFORE WE GET STARTED



This hack is fairly complicated for an introductory case modification. It includes a number of customized features to create a machine that meets all of my particular requirements. You can follow the hack step by step, choose the particular steps that you want for your own hack, or simply use the chapter as a reference for creating your own case mod hack. Have fun!

Choosing Your Features: Why the Atari 2600?

One of the first and most important steps in a case modification, or fitting a PC into a custom case as we're doing here, is to decide on the goals of the system. Sure, you could spend a lot of money on all the best high-end components, but that would be overkill if you just want to use the system as a server, to watch DVDs, or to play videogames. Creating a custom system is akin to an artist starting with a blank canvas: the final product can be anything you want it to be. There are no rules to follow.

Before embarking on the journey of a case modification, you should spend serious time considering the aesthetic and artistic aspects of the finished product. Try to imagine how it will look on completion. Consider different colors, designs, and lighting placements before buying any supplies.

In this hack, I will fit a fully-featured PC system into an Atari 2600 case (see Figure 2.1). I am a historian and collector of retro videogame systems and enjoy playing many different games on many different systems. I also have a personal, emotional connection to the Atari 2600, which is probably related to the nostalgia of growing up with one. From an engineering perspective, the design of the Atari 2600 hardware is both simple and complex—yin and yang, so to speak—and it has enticed me for many years. Since I want to retain as much of the original look and feel of the Atari system as I can, I will be using part of the original Atari circuitry, which contains the switches and connectors.

In 1977, when Atari introduced the Video Computer System (VCS)—later renamed the 2600—nobody, not even Atari, knew it would ultimately become a wild success and be the catalyst that would spawn the multibillion-dollar gaming industry we know today. It was one of the first generation of videogame systems that was not hardwired to play a certain set of games; today it is recognized around the world as *the* classic gaming system. (See Chapter 8, “Atari 2600,” for hacks and modifications for the Atari 2600 console.)

Figure 2.1 Original Atari Video Computer System, Model CX2600A



The goals of my Atari 2600PC creation are twofold. First, having recently relocated, I haven't had a chance to bring my collection of dozens of videogame consoles and vintage computer systems with me. So, I want this system to serve as an all-purpose videogame and computer emulator. This way, I can enjoy all the old consoles without needing the physical hardware and software (cartridges, floppy disks, tapes, and the like). The most important thing to me is to be able to play Atari 2600 and arcade games, although emulators exist for just about every videogame console; some more recognizable names include the Atari 5200, Atari 7800, RCA Studio II, Channel F, Nintendo NES, Nintendo Virtual Boy, Sega Master System, Sega Genesis, and even the more current consoles. The most stable emulators currently seem to run on Windows 2000/XP, and the additional hardware components I'll be adding are also supported in Windows, so picking the operating system was a fairly easy choice.

A subset of this first goal is to be able to use the original Atari game controllers with my system, instead of having to use the keyboard or mouse with the emulators. This is possible thanks to a new device known as the Stelladaptor, which we'll discuss later in the hack.

Second, I wanted a way to play DVDs (and, as a subset of that, CDs) on my TV in my living room. I admit that I don't have a standalone DVD player and most of the movies I own are still on VHS. But I'd rather hack together a custom system than just buy an off-the-shelf unit.

Knowing what features and functionality I want in the Atari 2600PC lets me properly plan for what I'll need to do throughout the hack, instead of working completely off the cuff, which could lead to problems as I try to fit everything together later.

The four switches on the top of the Atari (see Figure 2.2) will be used for the functions described in Table 2.1.

Figure 2.2 The Four Switches of the Standard Four-Switch Atari 2600 System



Table 2.1 Functionality of the Four Atari Switches, Before and After Hack

#	Original Type	Original Function	New Type	New Function	Notes
1	Toggle	Power on/off	Momentary	Power on/off	Replace with switch 3 (momentary) to work with motherboard
2	Toggle	TV type: Color/BW	—	Unused	
3	Momentary	Game select	Toggle	Unused	Replace with switch 1 (toggle)
4	Momentary	Game reset	—	Wireless mouse/keyboard connect	

There are six connections on the back of the four-switch Atari 2600 (see Figure 2.3). Their functions, before and after the hack, are listed in Table 2.2.

Figure 2.3 The Back Panel of the Standard Four-Switch Atari 2600 System

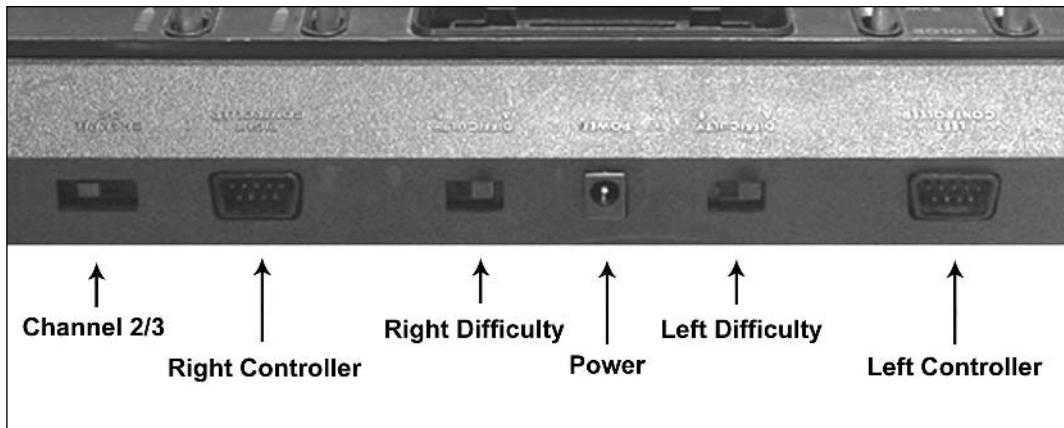


Table 2.2 Functionality of the Six Connections on the Atari's Back Panel, Before and After Hack

#	Type	Original Function	New Function	Notes
1	Slide	Channel: 2/3	Unused	
2	DB-9	Right controller	Player 2 controller	Interface with Stelladaptor
3	Slide	Right difficulty: A/B	Unused	
4	Jack	Power	Power	Replace with proper PC power supply jack
5	Slide	Left difficulty: A/B	Unused	
6	DB-9	Left controller	Player 1 controller	Interface with Stelladaptor



Preparing for the Hack

For this hack, we need to purchase lots of pieces, which requires a substantial commitment of funding. On completion, the total financial damage for my particular configuration of Atari 2600PC case modification was a little over \$700. Also, prepare to spend a significant amount of time fabricating and hacking components to get them to fit in the case. Basic soldering and desoldering skills are necessary (for an introduction to soldering, see Appendix A, “Electrical Engineering Basics”). The components required for the project are listed in Table 2.3, and most of them are shown in Figures 2.4, 2.5, and 2.6.

NEED TO KNOW ... YOUR WORKSPACE



Since the acts of painting, drilling, and cutting are very messy, it is imperative that you prepare your workspace by setting down newspapers or a drop cloth. If you live in an area where you can perform the major preparation outside, cleaning up will be much easier. It is also recommended (but not required) that you invest in a small handheld vacuum cleaner to clean up metal or plastic filings that have a tendency to accumulate during case modifications. The last thing you want is the interior of your case to be contaminated with these filings!

Table 2.3 Components for the Atari 2600PC Case Mod

Component	Notes
Atari VCS model CX2600A	Four-switch wood-grain version
VIA Technologies EPIA Nehemiah M10000 1GHz motherboard	See Figure 2.4. Can use other M-series suitable for your application. Available in most computer stores and online from places such as www.accupc.com , www.linitx.com , and www.mini-itx.com
PC2100 DRAM	Micron DDR 512MB, 266MHz, 184-pin
2.5-inch laptop hard drive	Fujitsu MHS2060AT, 60GB
40-pin to 44-pin 2.5-inch laptop IDE hard drive cable adapter	For connecting a 2.5-inch laptop hard drive to standard PC motherboard connectors
Slim CD-RW/DVD combo drive	Samsung SN-324
Slim-to-standard ATAPI/IDE adapter	For connecting a slim CD drive to standard PC motherboard connectors
PW-70 ATX Power Supply Module	70W, 12V DC-to-DC cableless converter for EPIA-M motherboards, available from iTuner (http://store.ituner.com/ituner)
AC-DC switching power adapter	12V, 5A power adapter for PW-70 Power Supply Module, available from iTuner (http://store.ituner.com/ituner)
ATX Power Extension Cable	Extension cable for use between EPIA-M motherboard and ATX Power Supply Module
Stelladaptor Atari 2600 Controller-to-USB interface (2)	Designed by Pixels Past and available exclusively from AtariAge (www.atariage.com)
DB9 joystick extension cable (2) (optional)	
USB four-port mini-hub (optional)	Needed only if using more than two internal USB accessories
Wireless keyboard and mouse (optional)	Logitech Cordless Access Duo Optical

Continued

Table 2.3 Components for the Atari 2600PC Case Mod

Component	Notes
802.11b wireless USB adapter (optional)	D-Link DWL-122
Type A male-to-female USB cable extender (optional)	Needed only with 802.11b wireless USB adapter
MPC II CD-ROM audio cable	
5.25- to 3.5-inch drive power adapter cable	
5.25-inch drive power cable Y-splitter	
2.1mm ID, 5.5mm OD PCB-mount power jack	Digi-Key #CP-202A-ND
1-inch-wide double-sided foam tape	3M #4011, exterior mounting, super-strong
3/4-inch-wide galvanized hanger strap	Two to three feet; also known as pipe support or plumber's strapping tape
Spool of 22–26AWG wire	Solid or stranded
6-32, 3/4-inch threaded standoff (2)	Aluminum or plastic with 1/4-inch threaded post (for motherboard mounting)
6-32 nut (5)	With optional lock-washer (for CD-ROM drive and motherboard mounting)
6-32, 3/8-inch screw (5)	

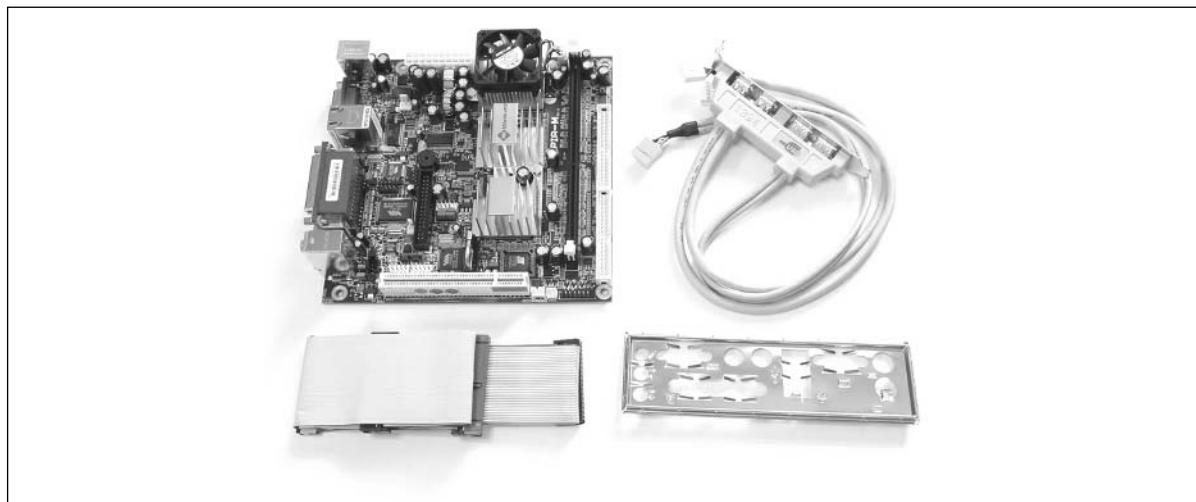
Figure 2.4 VIA Technologies EPIA Nehemiah M10000 Motherboard with Included Accessories

Figure 2.5 Most of the External Components Required for the Atari 2600PC Case Mod**Figure 2.6** The Logitech Cordless Access Duo Optical Keyboard, Mouse, and Receiver

The necessary tools you will need are:

- Phillips screwdriver (Regular and jeweler's size)
- Flathead screwdriver (Jeweler's size)
- Dremel tool with cutting discs (also called cut-off wheels)
- Drill with 9/64-inch drill bit

- X-ACTO/hobby knife
- Soldering iron
- Solder sucker or desoldering bulb
- Needlenose pliers
- Wire cutters/wire snips
- Small flat file
- Gorilla Glue
- Hot glue gun
- Liquid hand soap
- Small metal scrub brush or toothbrush
- Towel and washcloth
- Protective gear (goggles, mask, gloves)
- Compressed air (optional)

Even though these lists of parts and tools seem rather specific, all the items are quite common and generic in nature and can be easily found online or at your local computer store. The actual speeds and capacities of the computer components are a personal preference; I chose midrange parts so my new PC wouldn't quickly become obsolete. All the industrial parts and tools are also commonly found at any hardware store.

NEED TO KNOW ... FINDING AN ATARI 2600



Once I decided I wanted to build the PC into an Atari 2600 case, I set my sights on acquiring a few cheap and/or nonworking Atari 2600 systems. I wanted to obtain the original Atari 2600 six-switch wood-grain model, but the first system I came upon at a flea market was a nice-looking Atari 2600A four-switch wood-grain model, which was the second version of the Atari that was released (and also the more common version). Two of the toggle switches (Left Difficulty and Right Difficulty) that were on the top of the six-switch model were moved to the back on the four-switch. The hack in this chapter uses a four-switch model, but the same hack can be done with a six-switch model, with minor variations in preparing the inside of the case.

During my frequent visits to flea markets (another obsession of mine), I picked up three more Atari 2600 systems—one working six-switch model for \$20 and two nonworking four-switch models for \$5 and \$1. It helped to have multiple systems in case I needed to replace a part or messed up the housing during the modification. It also gave me the option of choosing the system pieces that were in the best physical condition. The Atari system itself doesn't need to work, since we're just hacking some of the switches and connectors to work with the PC and we won't be using the actual Atari 2600 circuitry.

The key is that the case be in good physical condition and look nice. Also, I made sure that the switches on top (two toggle and two momentary for the four-switch and four toggle and two momentary for the six-switch) moved properly and that the connectors on the back of the system were not cracked or damaged in any way.

If you live in an area without decent flea markets or computer surplus stores, buying a used Atari 2600 online is the next best option. eBay (www.ebay.com) and Bidots! (www.bidots.com) are both good starting points. You shouldn't have much trouble finding a system for \$5 to \$50.

One of the most important things is to make sure that the components you purchase will actually fit inside your enclosure. If possible, obtain the dimensions of each part, such as the motherboard, power supply, hard drive, and additional accessories. You can get the dimensions from manufacturer data sheets or product review pages.

Since I decided to obtain as many parts as I could through mail order to save money, I traced out the dimensions of each device on paper to make templates. This would let me see how much area I had inside the case before I actually spent money on the parts. The templates didn't help me with the height of the parts, which many times weren't shown on the data sheets. I decided to take the risk, knowing that I might have to tweak things later on when it came time to shove everything into the case.

VIA Technologies (www.via.com.tw) makes a number of small, fully integrated PC motherboards based around the popular Mini-ITX form factor. The EPIA Nehemiah M10000 motherboard measures only 17 cm by 17 cm (6.7 inches per side) and packs all necessary peripherals into one single unit. The board contains a 1GHz Nehemiah C3 microprocessor (which is an Intel clone that performs comparably to a Pentium III in benchmark tests) and VIA's CLE266 and VT8235 core-logic chip set. Peripherals include a six-channel AC'97 codec, accompanying an S/PDIF (Sony/Philips digital interface) audio-output option; a dual-channel FireWire transceiver; four USB 2.0 ports; 100Mb/second Ethernet interface; and an NTSC/PAL TV-out encoder. The M10000 supports two ATA133 mass-storage channels, along with a floppy drive connection, and uses a single DDR-266 DIMM for RAM. A graphics accelerator on board the CLE266 handles 128-bit, two-dimensional and 64-bit, three-dimensional graphics acceleration and includes hardware support for various MPEG-2 decoding tasks. A single PCI slot is available for expansion (but if you're using this motherboard in a small form-factor case, you'll probably ignore this feature). A high-performance gaming PC it is not, but for general computing, watching movies, and playing videogame emulators, this board is a sure hit.

NEED TO KNOW ... JOE'S MOTHERBOARD EXPERIENCE



When selecting parts for the Atari 2600PC case mod, I initially had decided to go with VIA's EPIA ME6000 600MHz motherboard. I chose this over the faster 1GHz Nehemiah platform because it did not require active cooling, so there was no on-board fan. By going with a fan-less design, which uses a large heat sink to remove heat from the processor that prevents it from overheating and getting damaged, the system would run silently (a welcome feature in this world of noisy computers and appliances). So, the tradeoff was slower and silent versus faster and louder operation.

When I received the package from accupc.com, the retail box said “EPIA ME6000,” just what I ordered. But when I opened the box, I noticed that the motherboard had a fan on it. The ME6000 was supposed to be fanless. Worried, I searched for some clues and found a small EPIA M10000 sticker on the parallel port connector on the back end of the motherboard. Lo and behold, I had been shipped the incorrect motherboard—they sent me a VIA EPIA Nehemiah M10000.

I was a little skeptical of using this board because I really wanted to have a silent PC and I didn’t know if the added height of the fan would prevent me from fitting everything into the Atari 2600 case. However, I’d be willing to overlook the small amount of sound (the fan is rated at a very low 24dB) for a whopping increase in processing power. Since I’ll be using this computer while the TV is on, the small amount of noise from the fan shouldn’t be a problem. I also usually leave my windows open in the living room, so I have some ambient noise from outside as well. Everything happens for a reason, as they say, and it turns out that the M10000 motherboard worked just fine and I’m pleased with the performance.

The 2.5-inch laptop-size hard drives are much smaller and draw less power than their 3.5-inch counterparts, but they are also more expensive. Judging by the placement of the templates inside the case, I wouldn’t have enough room for a regular-sized 3.5-inch laptop drive (which is actually more like 4 inches wide and 5.8-inches long once its metal enclosure is put in place). I decided on a Fujitsu 60GB MHS2060AT 4200RPM 2.5-inch hard drive, which will be plenty of storage for my system. If you decide to go with a 3.5-inch drive, remember to take into account the drive’s additional power consumption when you’re selecting a power supply unit.

Because of the space constraints inside the case, USB will be my primary interface to external devices. The EPIA M-series chipset can support six USB ports, but only four are actually usable (it is unknown why the other two USB ports are not brought out onto the motherboard). There are two ports on the external connector panel and two available from a 2×8 header on the motherboard. With the motherboard, VIA supplies a backplane attachment that connects to the header and provides those two ports. All my peripherals will be mounted internally to the Atari 2600 case so that they aren’t visible to the outside world.

Performing the Hack

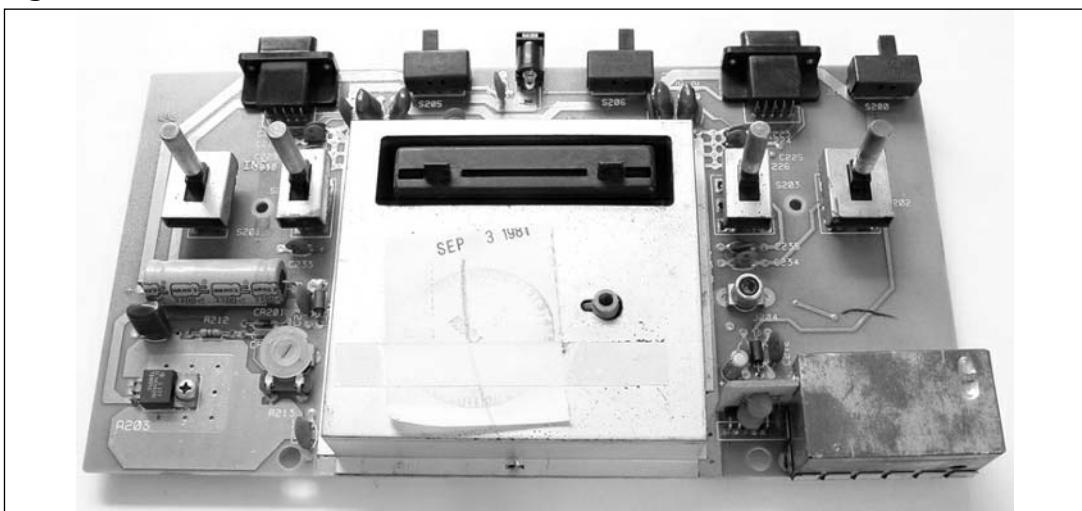
With all your parts and tools organized and ready, we can begin the actual hack. We’ll first prepare all the individual components and then fit them all together.

Opening the Case

1. To begin, remove the screws from the bottom of the 2600 using a standard Phillips screwdriver. There are four screws securing the case together. Note that the two bottom screws are longer, so you’ll want to be sure to insert these into the correct holes when you are reassembling the unit (see Figure 2.7).

Figure 2.7 The Underside of the Four-Switch Atari 2600

- Once you have removed the screws, pull the two halves of the case apart, unplug the RF cable from the board, and remove the circuit board from the case. Your system should resemble Figure 2.8.

Figure 2.8 Atari 2600 Circuit Board with RF Shield Attached

With the circuit board removed, you might want to use a can of compressed air to remove any clumps of dust on the board and inside the plastic housings of the case. Since we'll be using some of the original connections on this board (the four switches on the top and the joystick connectors on the back) later on in the hack, place the circuit board aside for now.

Cleaning the Case

Cleaning your case is the first step for a visually pleasing and successful case mod. Dirt and oils on the external case just don't look nice (and, if you plan to paint your case, will usually prevent paint from sticking).

With the case in two plastic halves, you can either clean the pieces in the dishwasher by running them through a *gentle* cycle, or you can hand-wash them in a sink or bathtub. Since I don't have a dishwasher, I used a bathtub and regular liquid hand soap.

1. First, rinse the case to get the large pieces of dust and dirt off it (see Figure 2.9).

Figure 2.9 Rinsing the Atari 2600's Plastic Housing (Top and Bottom Pieces)



2. Next, use a toothbrush lathered with the liquid hand soap to clean the detailed, hard-to-reach areas of the case, such as the ridges on the top of the case, the front panel, inside the cartridge slot, and the back connection holes (see Figure 2.10).

Figure 2.10 Scrubbing the Atari 2600's Plastic Housing



3. When the smaller areas are clean, use a damp washcloth lathered with soap to clean the larger areas of the case. When your case is nice and shiny (and reminds you of when you first took your Atari out of the box back in the early '80s), then you are ready to let it dry. Do *not* use a hair dryer or blow dryer to expedite the drying process; the hot air may cause the plastic of the case to warp. Simply pat the plastic halves with a towel or washcloth and then lay them down on a large towel to air dry (see Figure 2.11). After you're done cleaning, be sure to throw away the toothbrush or put it in a location with other tools so you don't accidentally brush your teeth with it in the future.

Figure 2.11 Letting the Shiny, Clean Atari 2600 Housing Dry



Mocking Up the Design

Before starting the modifications to the Atari case to hold the computer components, we'll connect all the components together outside the case to make sure they function properly and without conflict. By mocking up the system like this, it will help you figure out if you need extra components or what parts (if any) you can get rid of. Also, by installing and configuring all the necessary software now, you'll find that when it is time to stuff all the components into the case, the computer will be ready to power up right away. Mocking up the design also acts as an early "burn-in" test, so if there are any faulty components, they'll hopefully fail now instead of once everything is fitted into the case.

WARNING: HARDWARE HARM



Be sure to take proper antistatic precautions before working with electronic circuitry. All electronics should be handled only at a static-safe workstation with electrostatic discharge (ESD) mats and grounded wrist and ankle straps.

Attaching all the components together is pretty straightforward and will vary depending on what parts you are using. If you have ever assembled a PC from parts before, the following steps should look familiar:

1. Insert the DRAM into the DIMM slot.
2. Attach the power supply module to the ATX power supply connector.
3. Connect the CD/DVD to the primary IDE connector using ATA133 cable supplied with Mini-ITX (after connecting the slim-to-standard ATAPI/IDE adapter, as shown in Figure 2.12).
4. Connect the hard drive to the secondary IDE connector using the 2.5-inch laptop IDE hard drive cable adapter.
5. Attach the power cables to the hard drive and CD/DVD.
6. Attach the USB/FireWire backplane to the yellow USB connector on the motherboard marked with “USB3/4” on the top silkscreen.
7. Connect the USB hub and 802.11b wireless USB network interface card (NIC) to the USB ports on the backplane.
8. Connect the two Stelladaptors and the Logitech wireless mouse/keyboard receiver to the USB hub.
9. Attach a standard monitor, keyboard, and mouse.
10. Connect the DC power supply to the power connector.

With all your components connected, your setup should resemble the one shown in Figure 2.13.

Figure 2.12 Attaching the Adapter to the Back of the CD/DVD Combo Drive

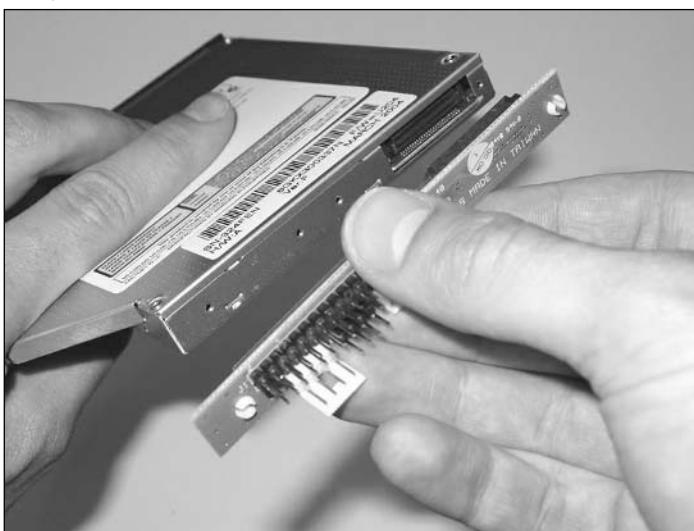
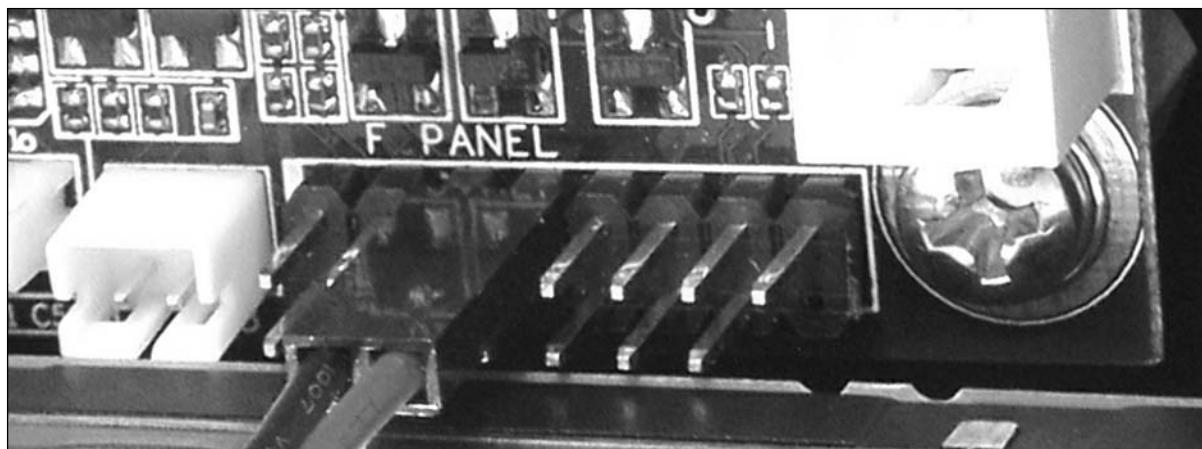


Figure 2.13 Connecting All the System Components



To turn on the computer, you will need to use a jumper or screwdriver to momentarily short pins 6 and 8 of the “F PANEL” connector located near the PCI connector (see Figure 2.14). The pinout to the F PANEL connector is shown in the user’s manual provided with the motherboard. When we eventually fit the computer into the Atari case, those two pins will be soldered to the momentary Power On/Off switch on our control panel.

Figure 2.14 The F PANEL Header to Turn On the Computer; The Connector Is on Pins 6 and

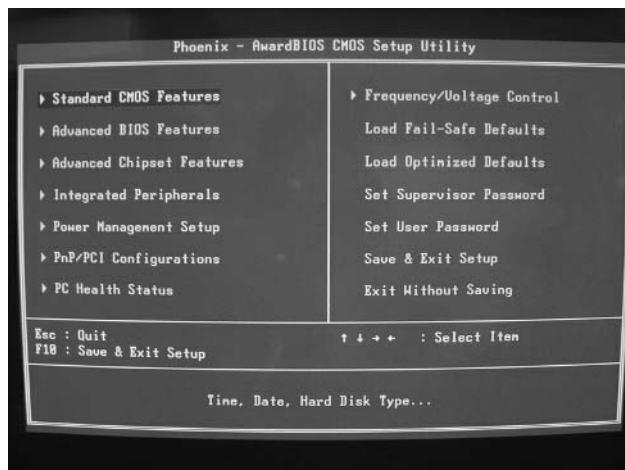


If the computer successfully powers on, the CPU fan will start spinning and you may hear your other devices coming to life. If the system does not start, immediately remove the power supply connection and recheck your connections to make sure they are all in the right place.

Configuring the BIOS

To enter the VIA BIOS configuration screen, hold down the Del key on your keyboard as soon as you power up the system. You will briefly see the EPIA logo and then be prompted with the screen for the Phoenix—AwardBIOS CMOS Setup Utility (see Figure 2.15). You might want to check the VIA Web site (www.via.com.tw) for the latest BIOS revision before you get started. From the BIOS menu, you can set the time, ensure that the motherboard is detecting your devices, and configure the motherboard for your particular specifications.

Figure 2.15 The VIA EPIA-M BIOS Screen



For my configuration, I changed the following settings, though your changes could vary:

- Standard CMOS Features
 - Date
 - Time
 - Drive A = NONE
- Advanced CMOS Features
 - First Boot Device = CDROM
 - Second Boot Device = HDD-0
 - Third Boot Device = DISABLED
 - Boot Other Device = DISABLED
 - Display Full Screen Logo = DISABLED
- Integrated Peripherals
 - Super IO Device
 - Onboard FDC Controller = DISABLED
 - Onboard Lan Boot ROM = DISABLED
- Power Management Setup (many of these settings will be overridden by ACPI-aware operating systems such as Windows 98/98SE/ME/2000/XP)
 - Power Off by PWRBTN = Delay 4 Sec
 - Peripherals Activities
 - PS2KB Wakeup from S3/S4/S5 = Ctrl+F1
 - PNP OS Installed = YES

With the BIOS configuration complete, insert the installation CD for your desired operating system, save the BIOS settings, and reboot the machine.

Installing Software

The process of configuring Windows and installing software applications are not covered in depth in this chapter. The applications you choose to install depend on how you intend to use the system.

For my system, I split the 60GB drive into two partitions. The C: drive, aptly named *Boring*, is 10GB and will hold the Windows OS and all applications. The E: drive, aptly named *Fun*, is approximately 50GB and will be used to store my movies, emulators, and game images.

As a guideline, I installed the software in the following order:

- Windows XP Professional
- Windows XP Update: Service Pack 1a, critical updates and patches
- VIA drivers (again, check the VIA Web site for the latest versions)
- D-Link DWL-122 802.11 USB adapter drivers
- Logitech Cordless Keyboard & Mouse drivers
- Nero 5 Burning Rom
- PowerDVD XP
- Emulators: MAME, z26, Atari800Win, MESS

With the cordless keyboard and mouse drivers installed, you can now remove the wired keyboard and mouse. When all the software is configured to your liking, the final step is to enable the TV output so you can attach the computer directly to a TV.

NEED TO KNOW ... USING THE EPIA-M WITH A TV



When you first configure the VIA EPIA-M motherboard, it will only boot via the VGA connector and not through the TV output. This is a known issue between the EPIA-M series and Windows (which overrides the motherboard's display settings in the BIOS). Apparently, this is not an issue with Linux-based systems.

Once Windows has loaded, you can enable the TV output using the Control Panel Display dialog box. With the TV output mode enabled, you can boot the computer using only a TV and without the monitor. You only have to select these settings once, because they will be saved for future use.

To have a readable display on a TV, you might want to configure Windows to use the High-Contrast Black display setting at 800×600 resolution and set the system fonts to Large (see Figure 2.16). The display settings aren't very conducive to performing actual *work* on the machine, but they are fine for selecting and controlling applications.

Figure 2.16 Windows XP and Software Installed onto the PC

Preparing the Control Panel

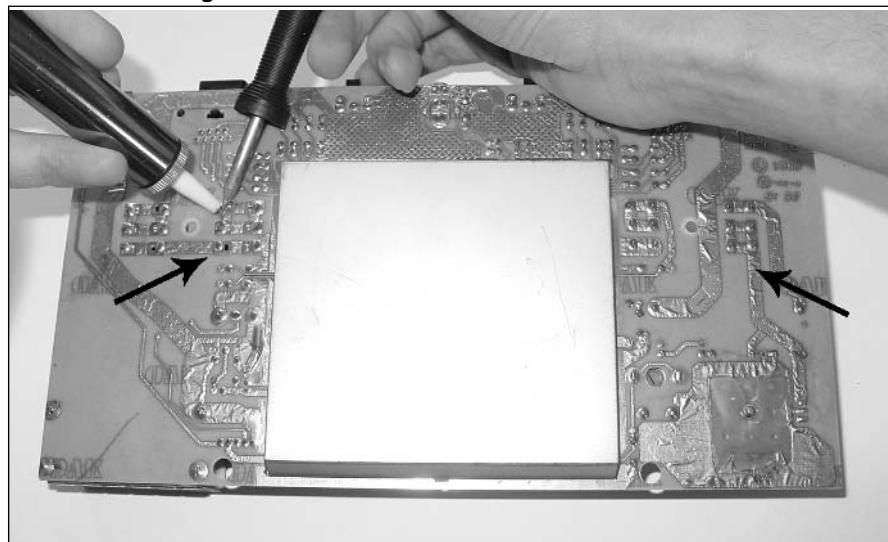
To prepare the original Atari circuit board for use in the Atari 2600PC, we first need to swap the toggle switch from the Power On/Off button (S201) with the momentary switch from the Game Select button (S203). This will give us the desired functionality that was described earlier, in Table 2.1. Having a spare Atari to use as parts can come in handy at this point, in case you damage the switches while removing them from the board.

WARNING: PERSONAL INJURY

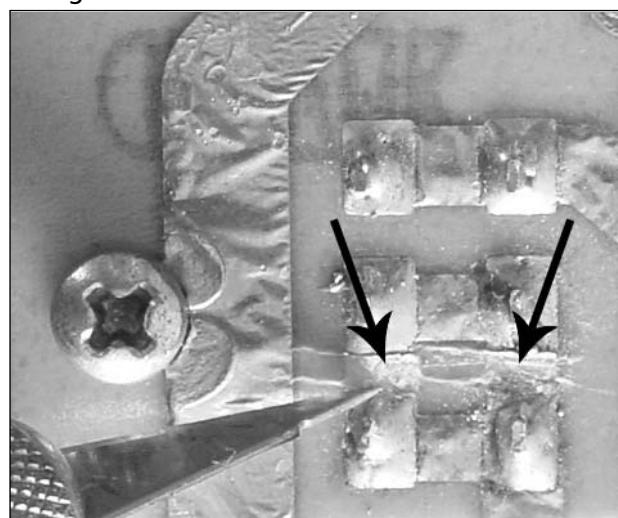


Always wear eye protection, respirator/mask, and other safety gear when using power tools.
Beware of hot tools and surfaces.

1. Before desoldering the switches, remove the four circular pads from the tops of each switch and place them aside. They are easily lost and you won't need them again until it is time to put the entire system back together. The two switches denoted with arrows in Figure 2.17 should be removed. There are six solder pads for each switch.

Figure 2.17 Removing Two Switches from the Atari Control Panel

2. Before soldering the new switches into place, you need to cut two traces on each side of the circuit board. On both the front and back sides of the switch, marked S201 on the top silkscreen (the right-most switch if you are looking at the bottom of the circuit board, as in Figure 2.17), use an X-ACTO knife to cut the two thick vertical traces that connect the two set of pads (denoted in Figure 2.18). This will allow a momentary switch to be used in place of the original toggle switch. After the cuts, you should have six discrete pads, with none of the pads connected to any other.

Figure 2.18 Removing the Vertical Traces from the Switch Pads

3. Now insert the toggle switch that you removed from S201 into the inner pads of the S203 footprint and solder it into place. The S203 momentary switch that is going into the S201 footprint is slightly too wide and won't fit in without modification (see Figure 2.19). Simply use needlenose pliers to bend the pins inward so they fit into the pads (see Figure 2.20); then you can solder the switch into place.

Figure 2.19 The Momentary Switch Leads Are Too Wide to Fit into the S201 Footprint

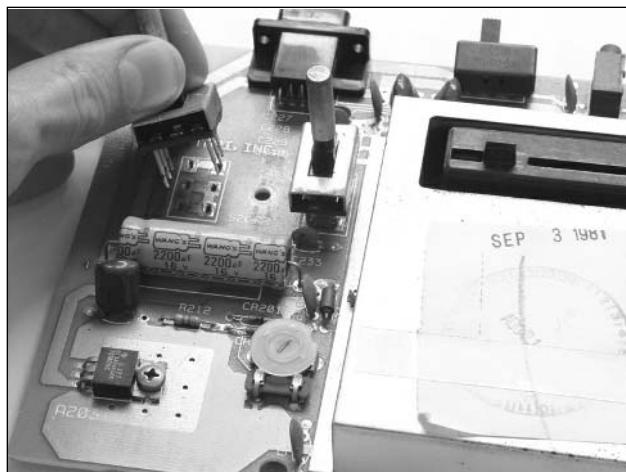
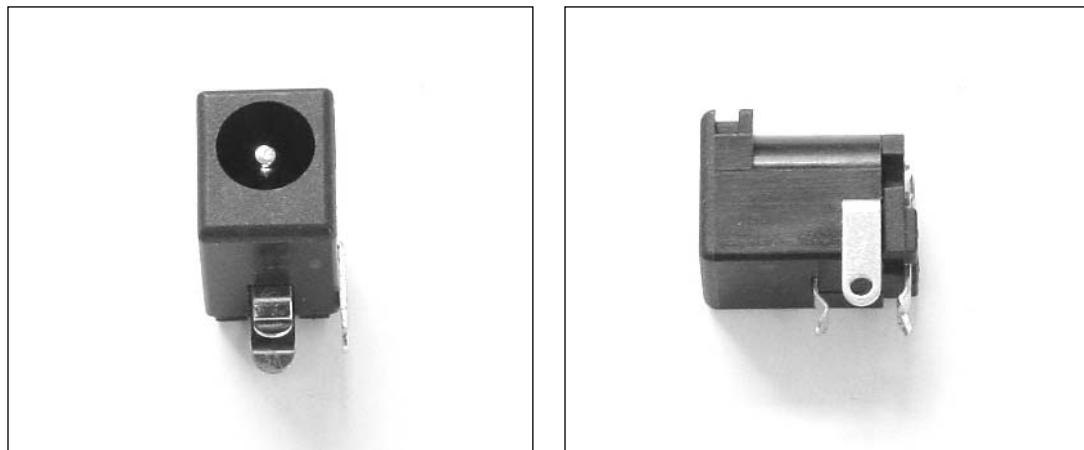


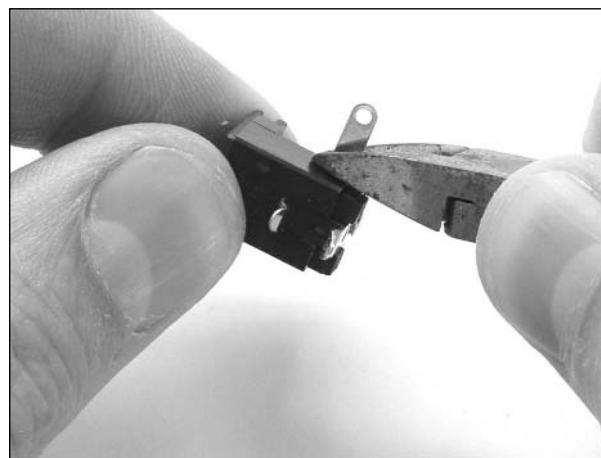
Figure 2.20 Bent Leads of the Momentary Switch to Fit into the S201 Footprint



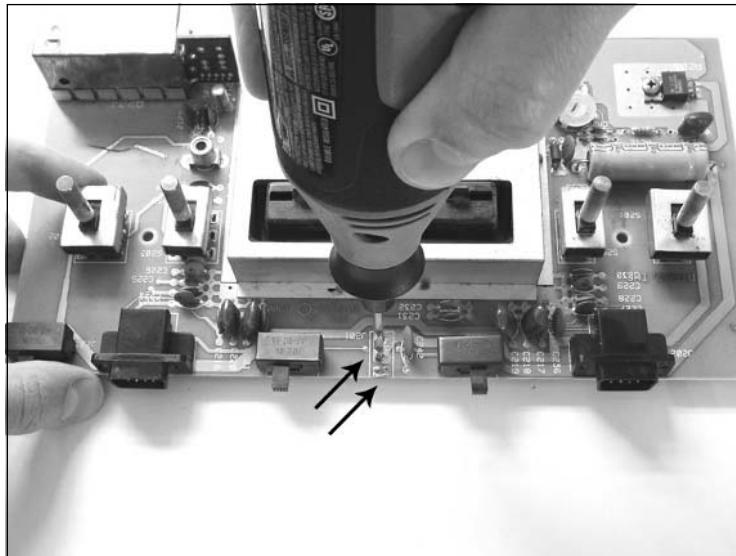
4. Next, we need to replace the power connector at J201 with the correct size for our DC power supply. In the case of the iTuner PW-70, a 2.1mm ID/5.5mm OD PCB mount power jack (Digi-Key #CP-202A-ND) works just fine (front and side views of the jack are shown in Figure 2.21).

Figure 2.21 New PCB Mount Power Jack, Front and Side Views

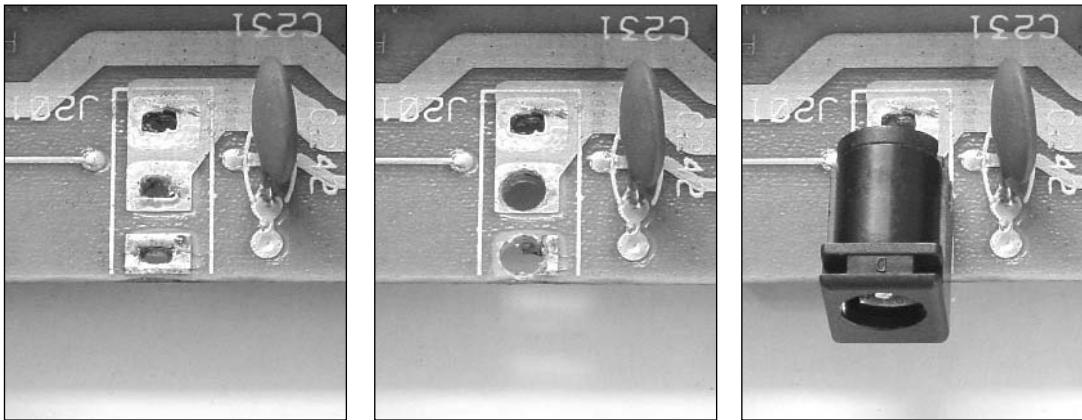
5. First, bend the side lead up and cut it completely off (see Figure 2.22).

Figure 2.22 Removing the Side Lead from the Power Jack

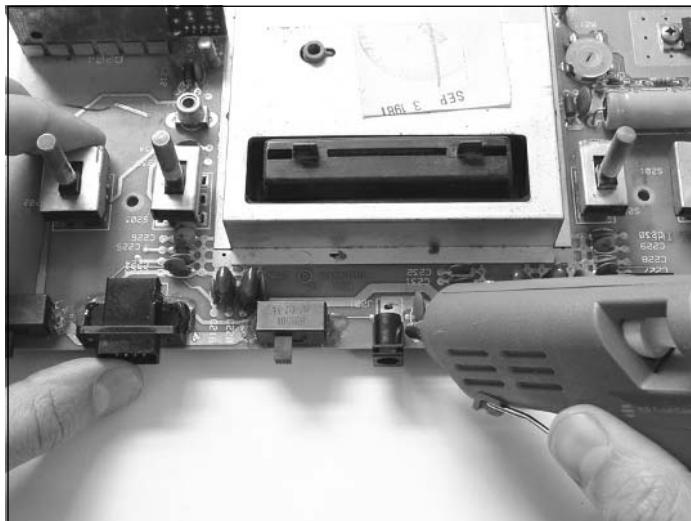
6. The holes on the circuit board for the original J201 are too narrow to fit the new power jack. For this reason, we need to increase the width of the holes. Using a 1/8-inch drill bit in a Dremel tool or drill, drill out the bottom two holes (when the board is rotated 180 degrees and the connectors are facing you, as shown in Figure 2.23).

Figure 2.23 Enlarging the Circuit Board Holes for J201

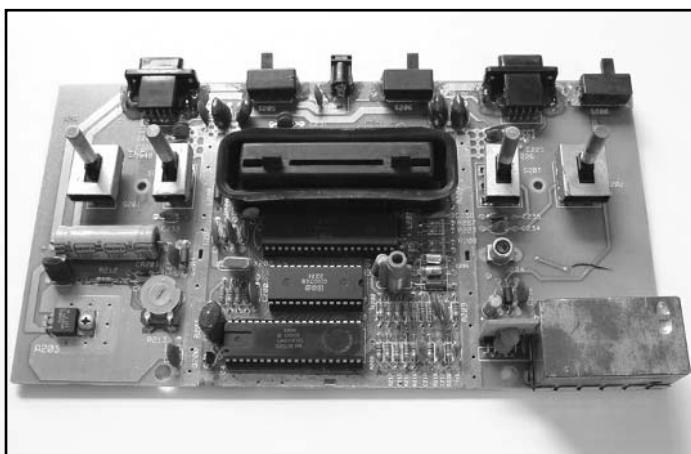
7. The power jack should now fit into the circuit board. When it does, solder it into place. Figure 2.24 shows J201's circuit board before drilling the larger holes, after drilling, and with the new power jack soldered on to the board.

Figure 2.24 Before, During, and After Placement of the New Power Jack into J201

8. Due to the old age and typically high wear of the components, it is recommended that you use some hot glue around the edges of all the connectors on the back to reinforce them and add some strain relief (see Figure 2.25).

Figure 2.25 Adding Hot Glue to the Connectors for Reinforcement

9. Before our next step, we need to remove the radio frequency (RF) shielding. To do this, use a pair of needlenose pliers to twist the small metal tabs along the edges of the RF shield so that tabs line up with the slots in the shield. You should then be able to pull the two halves of the shield apart from both sides of the circuit board. Once the RF shield is removed, the board should resemble the one shown in Figure 2.26. You can discard the shield, since we aren't using it for this hack.

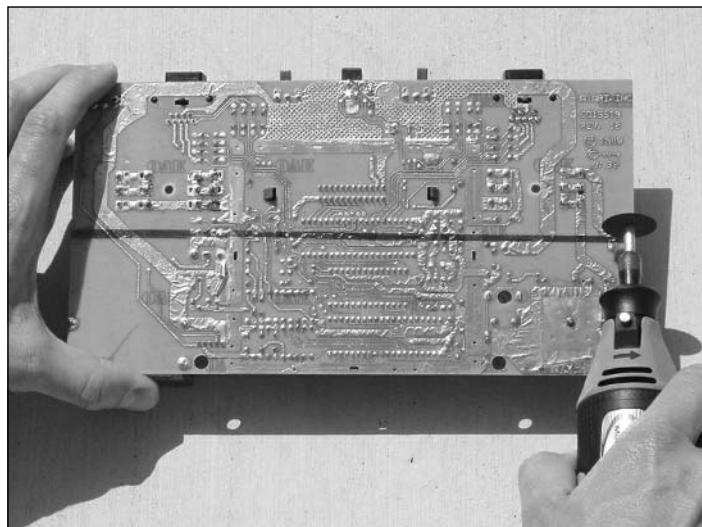
Figure 2.26 The Atari 2600 Circuit Board with RF Shield Removed

10. Next we'll use a Dremel tool to remove the unnecessary circuitry (which takes up valuable space within the Atari case). Before cutting, remove the integrated circuit (IC) located in the

socket near the C201 marking, directly underneath the cartridge connector. This will make it easier to cut the circuit board. We want to keep the connectors, switches, and cartridge connector in place while removing everything else.

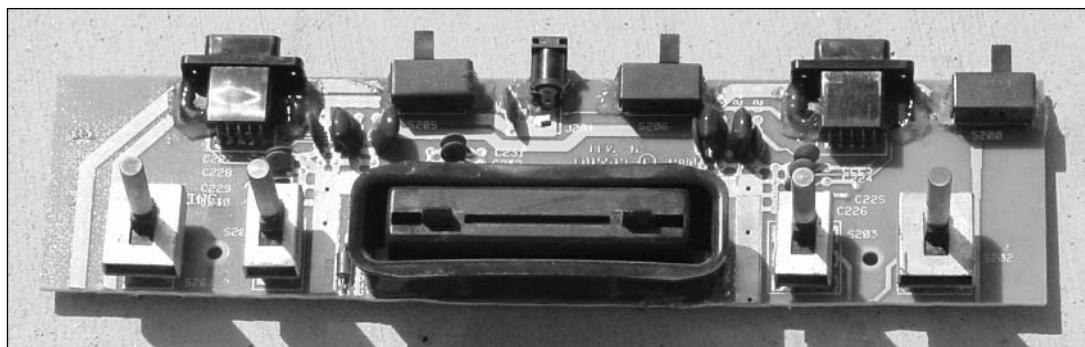
To mark the cut, simply flip the board over to the solder side (where there are no components) and draw a straight line across the board with a permanent marker. As shown in Figure 2.27, cut horizontally across the circuit board, about 2.5-inch down from the top of the board (where the connectors are located). Be sure to wear goggles during this step, because stray component pieces and fiberglass from the circuit board can lead to injury.

Figure 2.27 Cutting the Atari 2600 Circuit Board



You should now have a long, narrow control panel with all the necessary switches and connectors, as shown in Figure 2.28.

Figure 2.28 The Atari 2600PC Control Panel, Almost Done



11. Remove the rest of the extraneous circuit board material from the control panel to give us some additional vertical space within the Atari housing (which we'll need later on in the hack).
12. Then, use wire snips to remove all the discrete components (capacitors, resistors, and inductors) from the board. Anything that isn't a switch or a connector should be removed; if one of these old, unused components fails, the operation of the PC could possibly be affected.
13. Next, use the hand towel and liquid hand soap to clean the plastic cartridge connector, which is usually encrusted with decades of dirt. Be careful not to get soap or water into the switches or connectors on the back. The cartridge connector isn't used for our hack, but it is visible from the outside when the case is closed, so we want it to look nice.

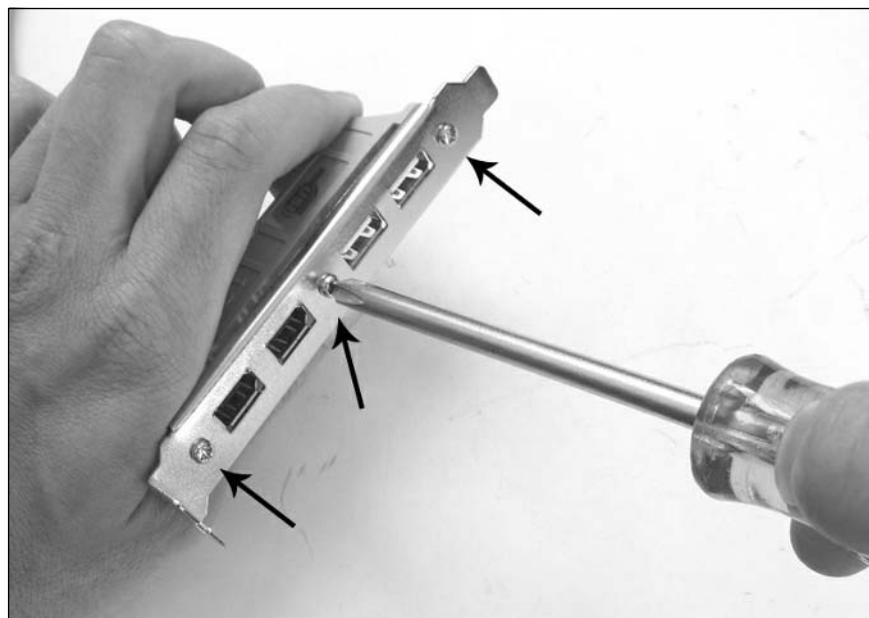
The control panel preparations are finally complete.

Preparing the USB/FireWire Backplane

The VIA EPIA M10000 motherboard comes with a USB/FireWire backplane, which contains the additional connectors for two FireWire (IEEE1394) and two USB ports. Space is hard to come by inside the Atari housing, so to reduce the amount of area that the connectors need, we will do a slight modification to remove the FireWire ports that we aren't using. If you have FireWire devices that you want to connect to the system, you might not want to perform this step.

1. First, remove the three screws that hold on the metal PCI expansion card rail, and throw the rail away (see Figure 2.29).

Figure 2.29 Removing the Metal Rail from the USB/FireWire Backplane



2. Next, use a Dremel tool to cut the backplane in half, removing the unneeded FireWire ports. Also, cut off the extraneous plastic from the other side of the USB ports. Be careful to not cut too closely to any of the connectors; you don't want to damage the connectors or wires (see Figure 2.30).

Figure 2.30 Removing the Unneeded Parts from the Backplane



When you're done, the backplane should resemble the one shown in Figure 2.31.

Figure 2.31 Modified USB Backplane



Preparing the Cordless Keyboard/Mouse Receiver

Next, let's remove all the unneeded plastic housings surrounding the receiver unit supplied with the Logitech Cordless Access Duo Optical Keyboard and Mouse (see Figure 2.32).

Figure 2.32 The Logitech Cordless Keyboard/Mouse Receiver Before Modification

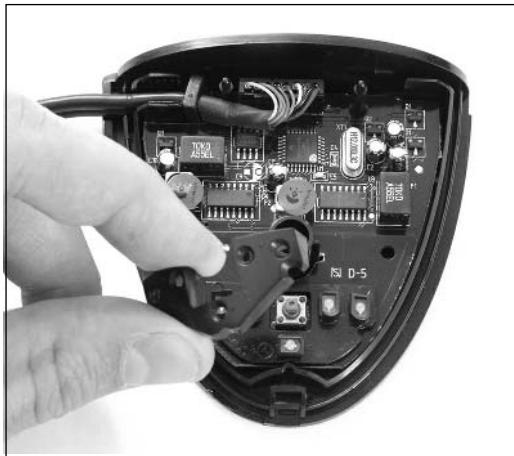


1. Open the unit by unscrewing the single screw on the back of the device and prying off the top half of the plastic housing (see Figure 2.33).

Figure 2.33 Opening the Receiver Housing

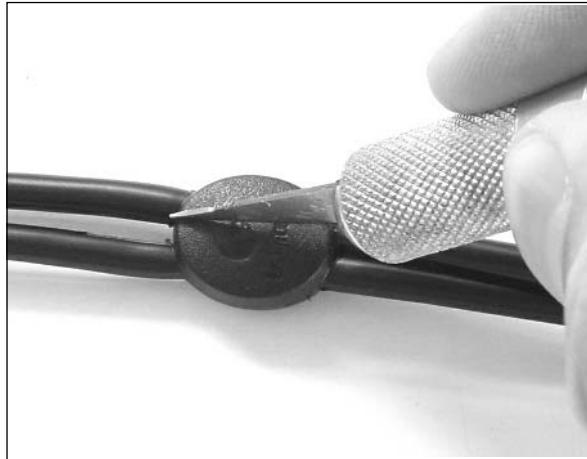


2. Next, remove the extra plastic piece that is sitting on top of the five LEDs at the front of the unit (see Figure 2.34). Then remove the whole circuit board from the bottom housing. The plastic housing and extra plastic piece can be discarded.

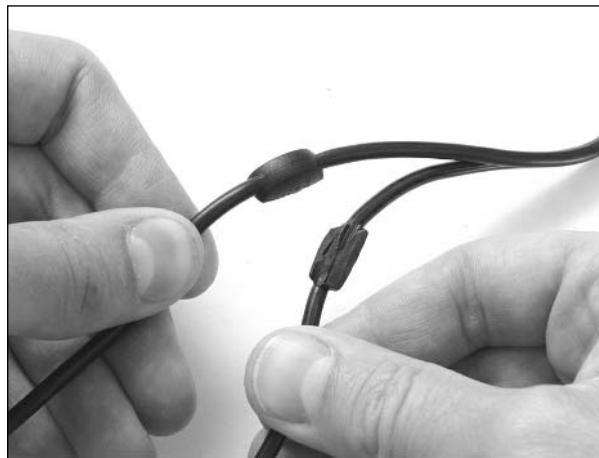
Figure 2.34 Removing the Unneeded Plastic Pieces

3. The receiver unit comes with both USB and PS/2 connectors. Because we'll only be using the USB connector, the PS/2 cable and connector can be removed, since it only takes up space.

First, use an X-ACTO knife to cut the plastic, oval-shaped Logitech stress relief in half (see Figure 2.35). Slicing through the middle of the stress relief will allow you to separate the two cables. The stress relief is located where the two separate USB and PS/2 cables join. Be careful not to slip let the knife slip and cut into either of the wires.

Figure 2.35 Cutting the Stress Relief in Half

4. Next, split the cable all way down to the circuit board by lightly pulling on each side of the cable (see Figure 2.36).

Figure 2.36 Splitting the USB/PS/2 Cable

5. Cut off the cable connected to the PS/2 connector (see Figure 2.37). Double-check which cable you are cutting before you do so. If you cut the wrong cable off the unit, it will be extremely difficult to repair. The USB connector and wire should be left in place (see Figure 2.38).

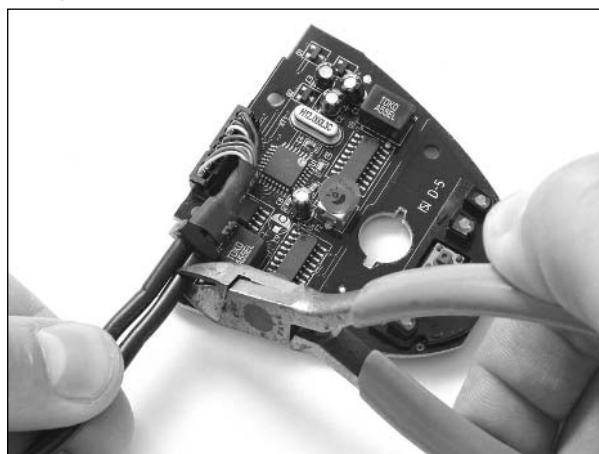
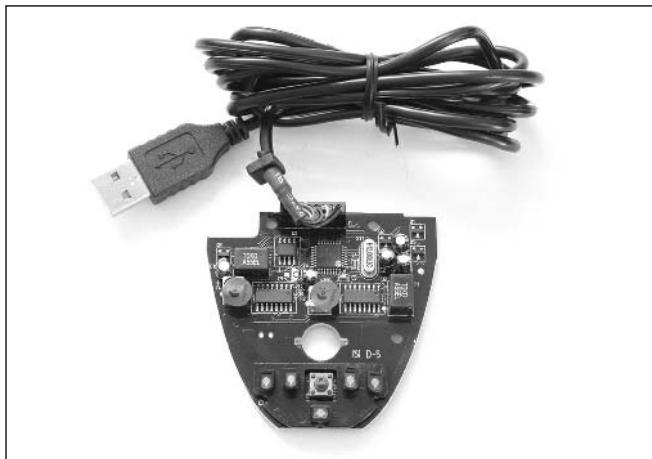
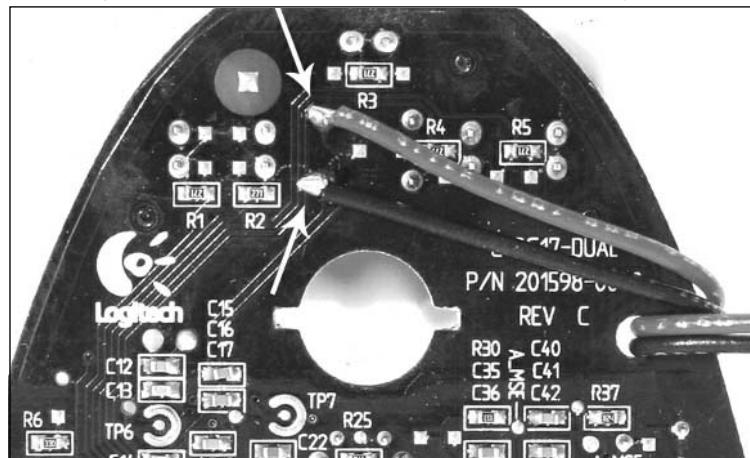
Figure 2.37 Removing the Unused PS/2 Connector

Figure 2.38 The USB-Only Receiver

6. Now we need to add an extension from the push-button switch on the receiver to a momentary switch on the control panel. This way, the Connect functionality (used to enable the wireless connection between the keyboard/mouse and the PC) can be activated without having to leave the wireless receiver accessible outside the case.

Simply solder an 18- to 20-inch length of 22-26AWG wire to each of the two pads denoted in Figure 2.39. The polarity of the wires does not matter. Wrap the wires through the hole on the right side of the circuit board, as shown in Figure 2.39, to act as a strain relief. The wires will be connected to the Atari control panel later in the hack.

Figure 2.39 Adding Wires to Extend the Switch Functionality

The modification of the cordless keyboard/mouse receiver is now complete.

Preparing the Stelladaptor 2600 Controller-to-USB Interfaces

The Stelladaptor 2600 Controller-to-USB Interface, designed by Pixels Past (www.pixelpast.com) and shown in Figure 2.40, allows the use of standard Atari 2600-compatible controllers, including joysticks, paddles, and driving controllers, with modern computers running Windows, Macintosh, or Linux operating systems.

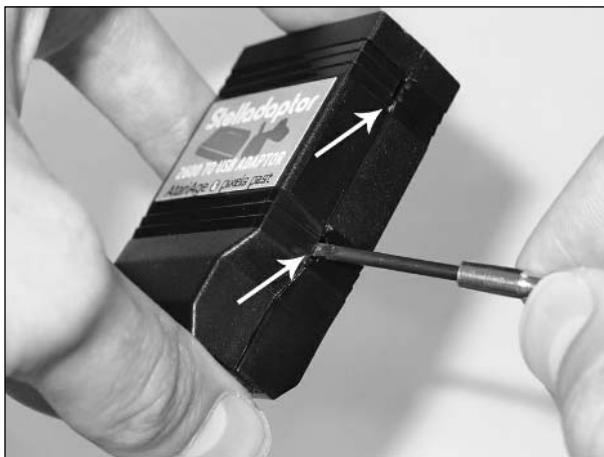
Figure 2.40 Stelladaptor Unit Before Modification



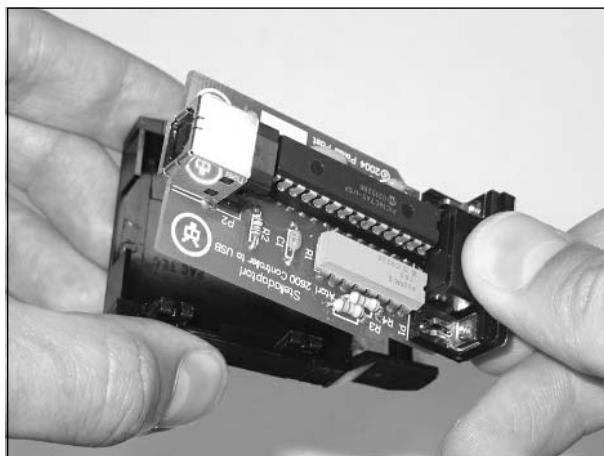
Any Atari 2600-compatible joystick, when plugged into a computer through the Stelladaptor, will behave as a normal joystick. This allows the use of classic Atari controllers with any software that will work with an eight-direction (left, right, up, down, and diagonals) digital joystick controller. In addition to joystick controllers, the Stelladaptor automatically recognizes the Atari 2600 paddle and driving controllers. Stelladaptor will work with any emulators that support standard USB game controllers. Paddle controllers will only work in emulators that allow configuration of analog USB controllers (such as MAME32 and MacMAME) or emulators that have been updated to directly support the Stelladaptor (such as z26).

Adding the Stelladaptor support into our Atari 2600 case mod will allow us to plug the original Atari controllers into the back of the 2600, just like in the old days, and use them with emulators running on the PC. It adds another point of authenticity to the case mod. Two Stelladaptor units are required: one for Player 1 and one for Player 2. The following instructions should be repeated for each unit.

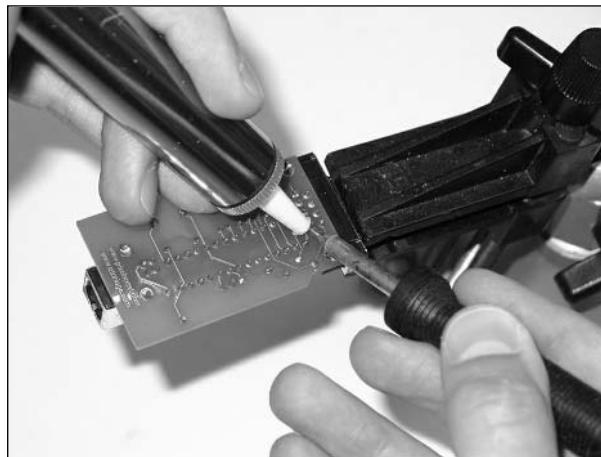
1. To prepare the Stelladaptor, you first need to remove the external plastic housing. You can do this using a small flathead jeweler's screwdriver to simply separate the two halves at the denoted latch points (see Figure 2.41).

Figure 2.41 Prying Open the Stelladaptor

When the device is opened, you can discard the plastic housing. You'll be left with a single circuit board, as shown in Figure 2.42.

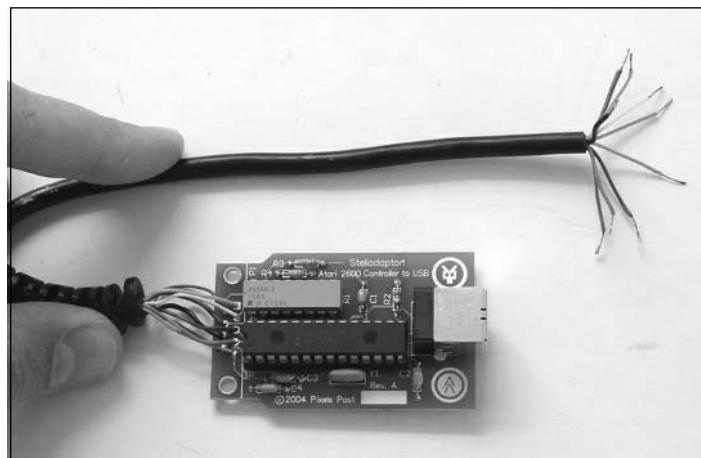
Figure 2.42 Removing the Stelladaptor Circuitry from Its Shell

2. To conserve space inside the Atari case, we will replace the DB9 connectors with nine discrete wires. Later on in the hack, those wires will be soldered directly to the pads of the original Atari's DB9 joystick connectors on the control panel. First, remove the DB9 connector from the Stelladaptor circuit board (see Figure 2.43). The DB9 is denoted as P1 on the silkscreen, but it is hard to miss (it is on the right side of Figure 2.42, underneath the thumb).

Figure 2.43 Desoldering the DB9 Connector from the Stelladaptor

3. Next, using nine 8-inch lengths of 22-26AWG wire, solder the wires into the DB9 pads on the Stelladaptor circuit board. If you have one available, use a DB9 Joystick Extension Cable with the ends cut off, which will give you nine wires all wrapped up inside a plastic sheath.

At this point, you should have two identical units that each resemble the one shown in Figure 2.44.

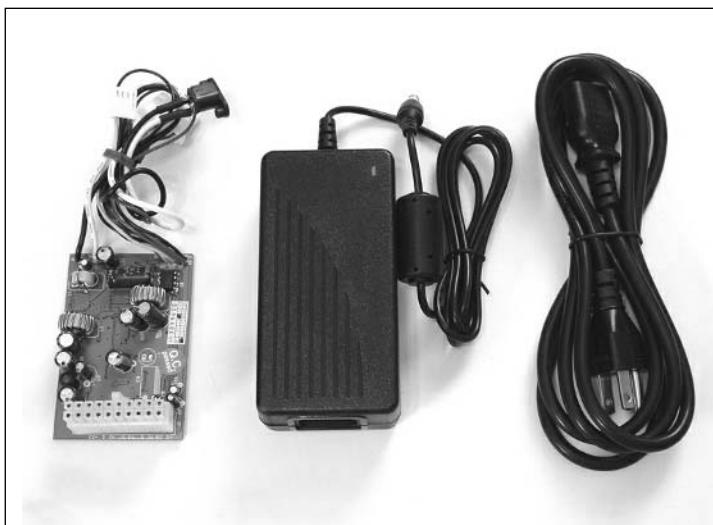
Figure 2.44 The Modified Stelladaptor with Wire Connections

The Stelladaptor modifications are complete.

Preparing the Power Supply Connector

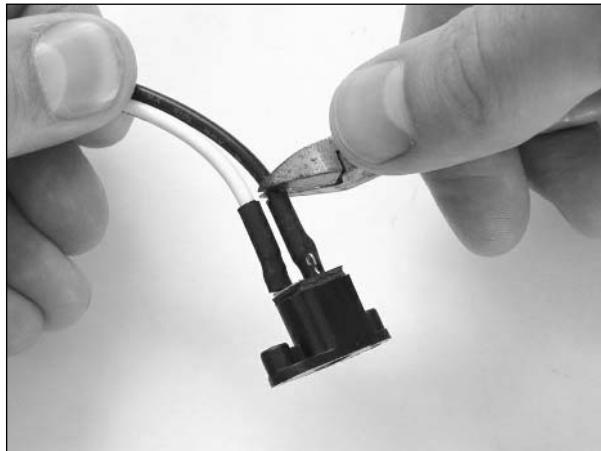
Two simple modifications to the iTuner PW70 ATX Power Supply Module (see Figure 2.45) are required to prepare it for the case mod.

Figure 2.45 iTuner PW70 ATX Power Supply Module and DC Converter



The black-and-white pair of wires on the PW70 serves as the 12VDC power supply input to the module. If a connector is provided with your module, cut it off as close to the connector as possible (see Figure 2.46). Later on in the hack, the white (+12VDC input) and black (GND) wires will be soldered directly to the power supply jack that is mounted to the control panel.

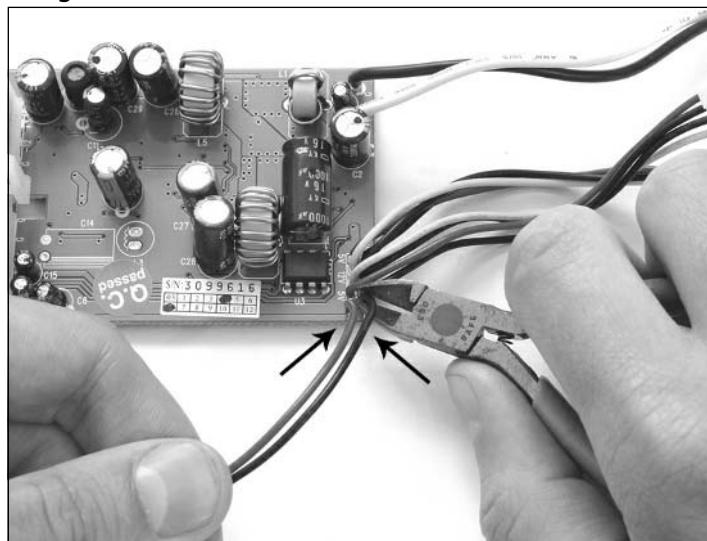
Figure 2.46 Removing the 12VDC Power Supply Input Connector from the PW70



The PW70 module also comes with a connector (the two-pin connector with red and black leads) that provides +5V DC for optional user applications. Since this connector isn't needed in the hack, it can be removed.

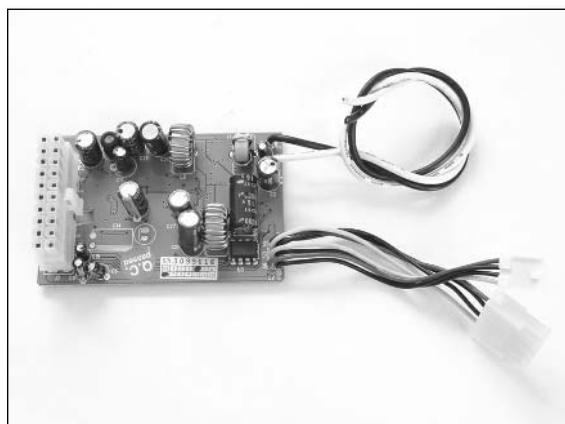
The red and black wires to cut are located on the lower-right corner of the module's circuit board, as denoted in Figure 2.47. Cut the leads off as close to the circuit board as possible (or desolder the red and black connections from the board) and place it aside for now. The 2-pin connector fits perfectly on the front panel (the F PANEL) header on the PC motherboard, so the connector and wires will be reused for our Power On/Off switch later in the hack.

Figure 2.47 Removing the Red-and-Black Two-Wire Connector from the PW70



The modified PW70 Power Supply Module should resemble the one shown in Figure 2.48.

Figure 2.48 The Modified PW70 Power Supply Module



Preparing the Mini-ITX Motherboard

Due to the lack of vertical clearance in the Atari housing, the ATX power connector on the motherboard needs to be modified so it will fit properly within the housing.

A modified ATX Power Extension Cable will be used to extend the power from the iTuner PW70 Power Supply Module onto the motherboard. This modification prevents us from taking advantage of the “cableless” solution of the PW70 (which is designed to plug directly into the ATX power connector on the motherboard), but it’s all in the name of hacking!

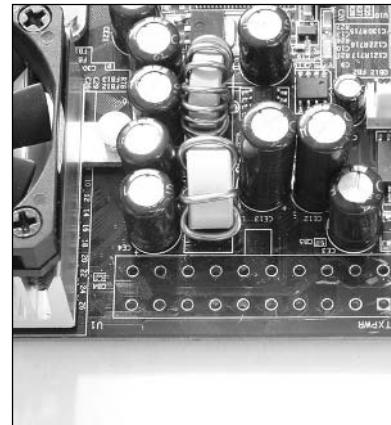
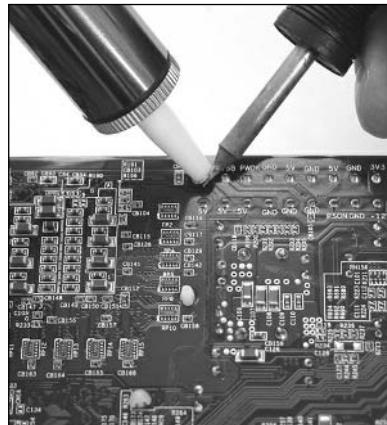
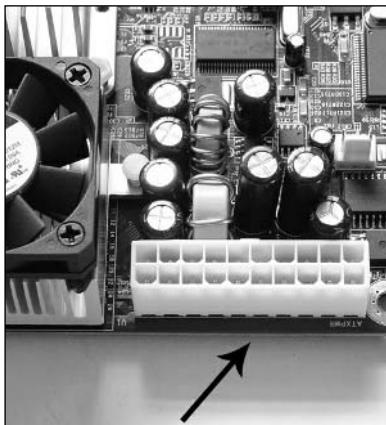
1. First, completely remove the plastic ATX power connector from the motherboard (denoted as ATXPWR on the silkscreen). By removing the connector, you’ll gain approximately half an inch of vertical clearance. Figure 2.49 shows the EPIA-M motherboard before the ATX power connector was removed, while the connector is being desoldered, and after the successful modification.

WARNING: HARDWARE HARM



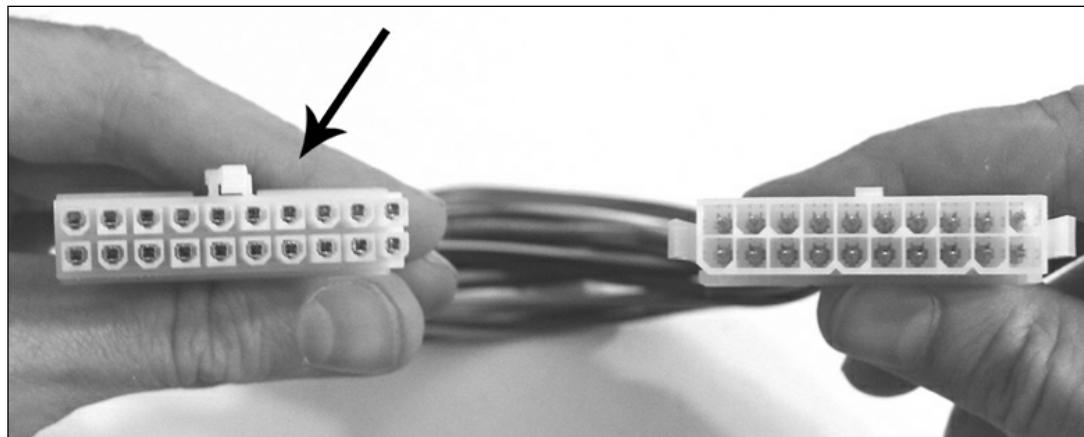
Be careful while removing the ATX power connector from the EPIA-M motherboard. The motherboard contains dozens of tiny, densely packed components. Any stress on the board could cause parts to come loose or become damaged. Also, take care not to scratch or damage any PCB traces.

Figure 2.49 Removing the ATX Power Connector from the Motherboard: Before, During, and After Modification



2. Next, remove the *male* connector from the ATX Extension Cable. This is the side that looks the same as the connector mounted on the PW70 Power Supply Module and is denoted in Figure 2.50. Each of the 20 wires will be soldered to the bottom of the Mini-ITX motherboard, giving the necessary vertical clearance on the top of the motherboard.

Figure 2.50 The ATX Extension Cable Before Modification; Arrow Denotes Male Connector to Be Removed



3. Once the connector is removed, strip about 1/4 inch of insulation from each wire. Twist the stranded wires at the end of each lead on the ATX Extension Cable and “tin” them with a small amount of solder (see Figure 2.51).

Figure 2.51 ATX Extension Cable Ready to Be Soldered to the Motherboard



Now the modified extension cable is ready to be inserted and soldered into the motherboard. Luckily, each connection is designated with its function on the bottom silkscreen of both the motherboard and the Power Supply Module, so it is easy to see where they connect (for example, +5V to +5V, GND to GND, -12V to -12V, and so on). Although the wire colors of the ATX Extension Cable may vary, mine were laid out in the following fashion (reading counter-clockwise starting with yellow at the upper right):

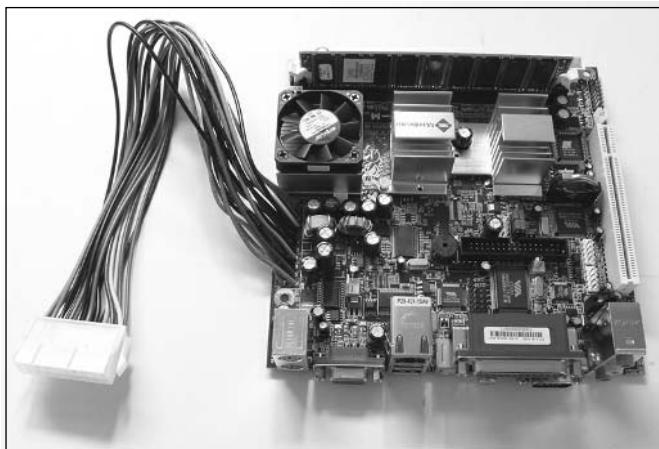
Row 1:

- Yellow = +12V
- Purple = 5VSB
- Grey = PWOK
- Black = GND
- Red = 5V
- Black = GND
- Red = 5V
- Black = GND
- Orange = 3.3V
- Orange = 3.3V

Row 2:

- Orange = 3.3V
- Blue = -12V
- Black = GND
- Green = PSON
- Black = GND
- Black = GND
- Black = GND
- White = -5V
- Red = 5V
- Red = 5V

When the extension cable is soldered into the motherboard, the modified EPIA-M motherboard should resemble the one shown in Figure 2.52.

Figure 2.52 The VIA EPIA-M Motherboard with Modified ATX Power Cable

Preparing the Housing

This section is the messiest and most time consuming, but when you're done, you'll have an Atari 2600 case waiting with open arms and ready to be stuffed full of PC components.

WARNING: PERSONAL INJURY

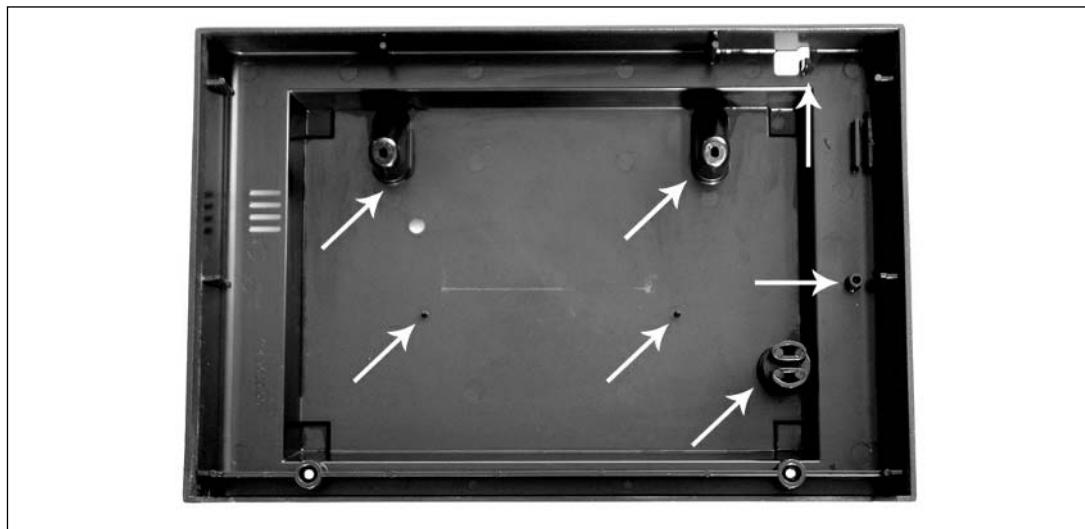


Always wear eye protection, respirator/mask, and other safety gear when using power tools.
Beware of hot tools and surfaces.

1. Figure 2.53 shows the Atari 2600 bottom housing with arrows pointing to the seven areas where plastic needs to be removed. Removing the unnecessary plastic will give us more space inside the case so we can fit all the parts inside.

Using the Dremel tool and cutting wheel, remove the two large recessed screw holes/struts located toward the back of the case. When you remove these pieces, there will be two oval holes going out through the bottom of the case. Next, remove the two thin posts located in front of those struts and the lone post located on the right edge of the case. Also, remove the wire stress relief post on the upper-right side of the case. Finally, remove the top section of the wire stress relief column on the lower right side of the case. Remove only the top two plastic ovals. The bottom cylinder should be left in place because it will act as the hard drive support later on.

Figure 2.53 Unmodified Atari 2600 Bottom Housing with Arrows Denoting Plastic to Remove



With the plastic pieces successfully removed, your case should resemble the one shown in Figure 2.54.

Figure 2.54 Atari 2600 Bottom Housing with Unnecessary Plastic Removed



2. Next, we'll prepare the case to hold the motherboard and the CD-ROM drive. Since the CD-ROM drive will be sitting below the motherboard, it is the first thing that needs to be mounted. The CD-ROM drive will be situated facing the front of the case, so it will open

straight toward you. Because the drive is slim, it fits perfectly under the lip of the bottom housing, doesn't interfere with the wood-grain bezel, and is fairly unobtrusive, so it won't be obvious that the Atari 2600 has been modified.

Align the drive in the center of the case and use a permanent marker to outline the rectangular area that will need to be removed from the front of the case (see Figure 2.55).

Figure 2.55 Marking the CD-ROM/DVD Drive Tray Location on the Atari 2600 Bottom Housing



3. Next, carefully Dremel out the marked area. It is easiest to do this from the inside of the case, slightly within your marked rectangle. Then, use a small file to flatten out the sides right up to edges of the markings. You should now have a nice rectangle under the front lip of the housing through which the CD-ROM tray can fit (see Figure 2.56).

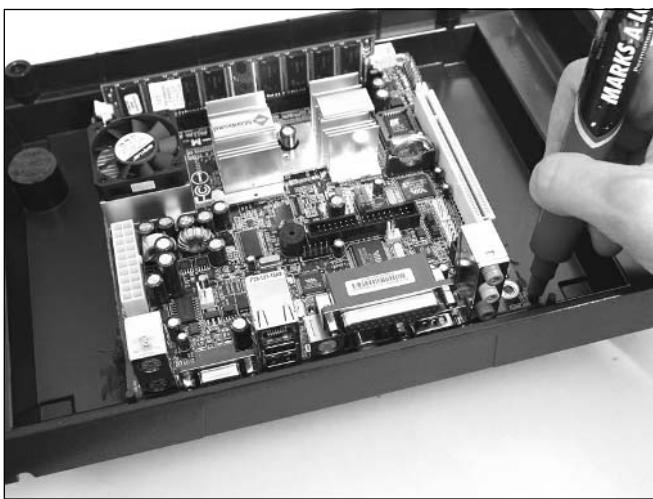
Figure 2.56 The Completed CD-ROM/DVD Drive Slot



The motherboard will be situated inside the housing with its connectors facing the back of the case. This way, any physical connections will be out of sight. In order for the motherboard's connector panel to be accessible, the back of the Atari housing will need to be modified. We'll prepare the bottom of the Atari housing first and work on mating the top housing later.

4. Place the motherboard inside the bottom housing and use a permanent marker to outline the rectangular area of the connector panel that will need to be removed from the back of the case (see Figure 2.57). Be sure that the CD-ROM drive is sitting in place first, because the motherboard will be sitting on top of it and will be lifted slightly off the bottom of the case. If you want to use the metal connector shield that is included with the Mini-ITX motherboard, be sure to attach it to the connectors before making your measurements. For my hack, I chose not to use it because I thought it took away from the simple aesthetics of the Atari 2600 case and just looked cheesy and of poor quality (the shield is made from a thin, stamped aluminum and can be seen at the bottom right of Figure 2.4). Without the shield, it might be easier for dust and debris to float into the back of the system and onto the motherboard, but it also could increase the much needed airflow inside the case. Using the connector shield or not comes down to a personal decision.

Figure 2.57 Marking the Motherboard Location on the Back of the Atari 2600 Bottom Housing



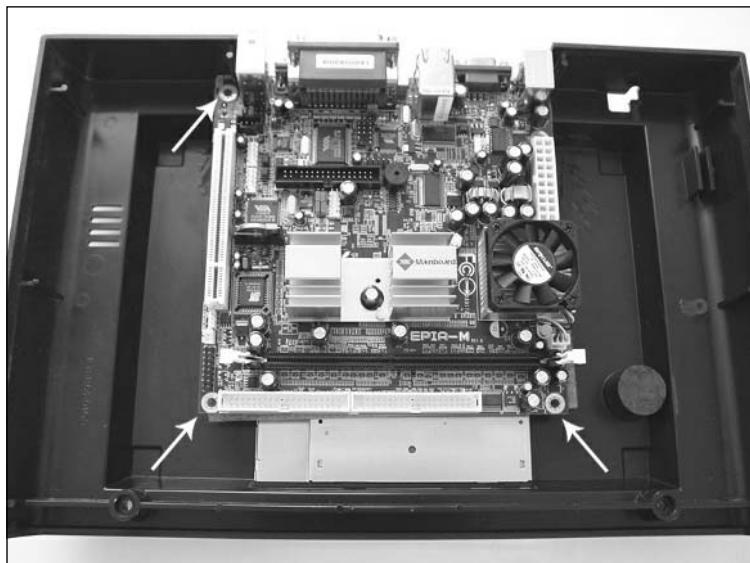
5. Next, carefully Dremel out the marked area (see Figure 2.58). It is easiest to do this from the inside of the case, slightly within your marked rectangle. Then, use a small file to flatten out the sides right up to the edges of the markings. Finally, cut a slot in the left support post (also denoted in Figure 2.58), which will enable the motherboard to slide all the way flush with the back of the case, making the connectors easier to access once the case is closed.

Figure 2.58 Marked Area for the Hole and Slot on the Back of the Atari 2600 Bottom Housing



6. To mount the motherboard to the bottom of the Atari case, you need to drill three holes (their locations are denoted in Figure 2.59). The diameter of the holes will vary depending on the diameter of your standoffs and screws, but I used a 9/64-inch drill bit for the 6-32 size hardware. Later on in the hack, the two screw holes on the front of the motherboard will be screwed into place with standoffs and the screw hole in the back corner will be fastened with a nut and bolt. Don't screw anything together quite yet.

Figure 2.59 Drilling Locations for Motherboard Mounting Holes



The completed bottom Atari 2600 housing should resemble the one shown in Figure 2.60. Now we can move on to the top half of the Atari housing.

Figure 2.60 The Completed, Modified Atari 2600 Bottom Housing



7. First, the original hole in the back of the case intended for the power jack needs to be enlarged. The new power jack that was soldered on during the control panel preparation is slightly larger than the original Atari one. Simply use a small, flat file to enlarge the square hole in the back of the case (see Figure 2.61) until the new power jack fits snugly through it (see Figure 2.62). Remove the control panel after the jack has been successfully enlarged; we'll screw it into place later on in the hack.

Figure 2.61 Enlarging the Case to Fit the New Power Jack

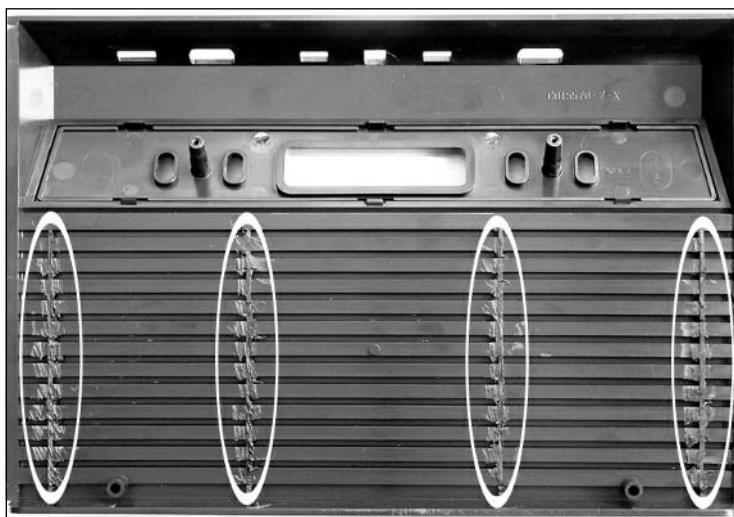


Figure 2.62 Checking to Make Sure the New Power Jack Fits Snugly



8. Next, Dremel the four “ribs” on the underside of the top housing (see Figure 2.63). They take up much needed vertical space, and the motherboard will not have enough room to fit inside unless they are removed.

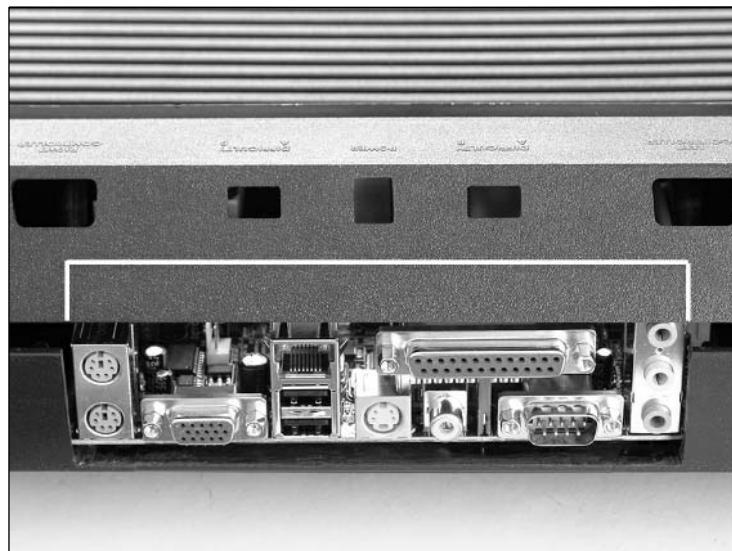
Figure 2.63 Removing the Ribs from the Underside of the Atari 2600 Top Housing



9. Place the CD-ROM drive and motherboard back into the bottom housing. Don’t mount them yet—this step is just for measurement purposes. Set the top half of the housing on top, as though you were putting the case back together. It won’t fit fully yet, because the connectors on the back are too high. Use a permanent marker to outline the rectangular area

that will need to be cut from the top housing in order for the whole connector panel to fit through the back. The rectangle will be approximately 9/16-inches high and 5.75-inches long, centered with the rectangle cut of the bottom housing, as shown in Figure 2.64.

Figure 2.64 Marked Area for the Connectors on the Back of the Atari 2600 Top Housing



10. Next, carefully dremel out the marked area. It is easiest to do this from the inside of the case, slightly within your marked rectangle. Then, use a small file to flatten out the sides right up to the edges of the markings.
11. The final step for preparing the top housing is to cut a large notch horizontally across the underside of the Atari 2600 top housing so that the DRAM can fit inside without hitting the top of the case. Without this modification, the DRAM is too high and will prevent the case from closing properly. The location of the notch may vary slightly depending on the variations of how you mount your motherboard. Using a Dremel tool, enlarge the slits on the underside of the case, approximately where the DRAM hits the top (you can try to stick your finger through the hole in the back of the case and feel around to where the DRAM is located). The locations are denoted in Figure 2.65. Take care not to remove too much plastic from the slits, since you might end up with cuts appearing on the top side of the housing.

Figure 2.65 The Completed, Modified Atari 2600 Top Housing, with Arrows Denoting Slots for DRAM



Putting It All Together

Now that all the preparations are completed, it's time to move on to cramming everything into the Atari case. Remember, this is a hack, so there is more than one "right" way to do things. If after following these directions something doesn't fit as you like, don't be afraid to experiment with other methods to place and mount the components. The case mod is ultimately a reflection of you and your personality.

WARNING: HARDWARE HARM



Be sure to take proper antistatic precautions before working with the electronic circuitry. All electronics should be handled only at a static-safe workstation with ESD mats and grounded wrist and ankle straps.

The CD-ROM Drive

1. First, we need to mount the CD-ROM drive to the bottom of the Atari housing. Cut an 8.5-inch length of the hanger strap and fold it around the top of the drive so it fits like a brace (see Figure 2.66). Try to make it as snug fitting as possible without creating undue stress on the top or sides of the drive.

Figure 2.66 The CD-ROM Drive with Hanger Strap “Drive Rail”



2. Using Gorilla Glue, glue the brace to the top of the drive (slightly moisten the top of the drive with some water before applying the glue). Clamp or weigh down the strap with a few heavy books and wait three or four hours for the glue to cure. Be careful that the books aren't so heavy that they warp the top of the drive. The hanger strap will act as our drive rail.
3. When the glue is dry, align the CD-ROM drive with the slot hole that you prepared earlier. Use a permanent marker to mark one hole on each side of the drive rail onto the bottom of the Atari case. Using the proper drill bit size based on the nut diameter you are using (I used a 9/64-inch bit for the 6-32 screw), simply drill out the two holes (see Figure 2.67).

Figure 2.67 Drilling Mounting Holes for the CD-ROM Drive

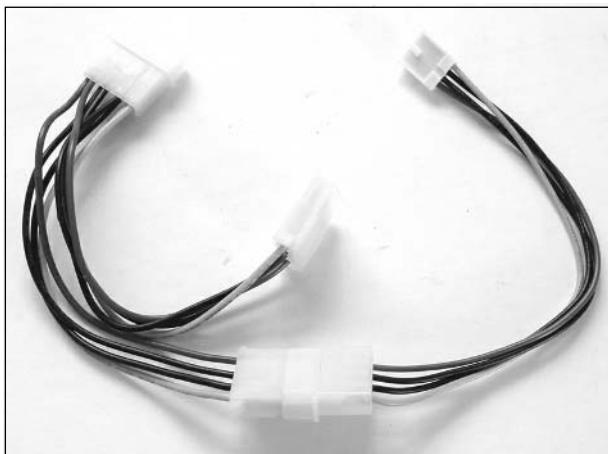


4. Next, insert the two screws from the underside of the Atari case and attach the nut on top of the drive rail. With the two screws in place, the CD-ROM should be securely mounted and should not wiggle or move around in any direction (see Figure 2.68). Take care to not over-tighten the screws, placing stress across the top of the CD-ROM drive, which could cause disc vibration or rattling when the entire system is put together.

Figure 2.68 The CD-ROM Drive Securely Mounted to the Atari 2600 Bottom Housing



5. Because the 3.5-inch drive PC power connector on the PW70 power Supply Module (which is used to connect to the slim CD-ROM drive) is too short and won't reach to the drive, we need to create an extension cable using the 5.25- to 3.5-inch drive power adapter cable and a 5.25-inch drive power cable Y-splitter (see Figure 2.69).
6. With the custom power cable extension, there is ample length to connect it between the Power Supply Module and the slim CD-ROM drive. Now connect the power, audio, and IDE cable to the back of the drive.

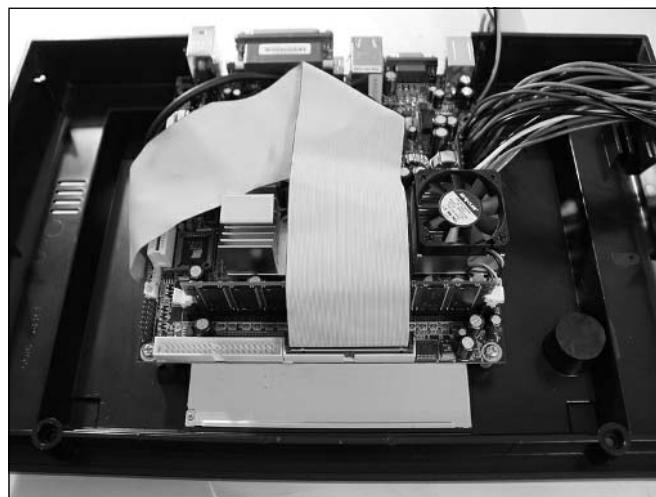
Figure 2.69 Custom Power Cable Extension

The Motherboard

1. Align the motherboard into the bottom of the Atari housing. The connectors should be facing out the back, as flush to the edge as possible, as shown in Figure 2.59. To secure the motherboard, first insert a screw from the bottom into the motherboard mounting hole next to the connectors on the back panel. Insert a nut from the top of the motherboard and tighten it into place. Next, place the two 3/4-inch standoffs underneath the two front mounting holes, insert the nuts from the top of the motherboard, and tighten them. From the underside of the case, insert a screw into each of the two holes and loosely screw them in.
2. Now, connect the ATA133 and audio cables from the CD-ROM onto the motherboard. Both cables should come out from underneath the left side of the motherboard. The audio cable should plug into the mating audio connector (labeled “CD IN” on the silkscreen). Flip the ATA133 cable over and connect it to the IDE connector marked “PRIMARY” on the silkscreen.

Your system should resemble the one shown in Figure 2.70.

Figure 2.70 The EPIA-M Motherboard Mounted into the Atari 2600

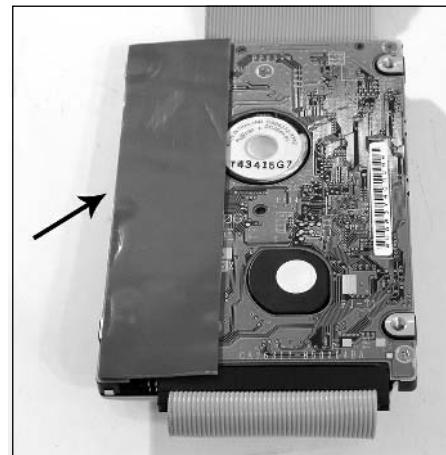


The Hard Drive

We'll mount the hard drive to the case with pieces of foam tape. The tape is extremely strong and will hold with typical use of the PC. It also serves double duty as a shock absorber. Velcro could also be used for easier removal of the hard drive, if you're planning to upgrade at a later date.

1. First, attach the IDE cable to the back of the drive. Then, attach a 3.75-inch length of foam tape to the bottom left side of the hard drive, as shown in Figure 2.71.

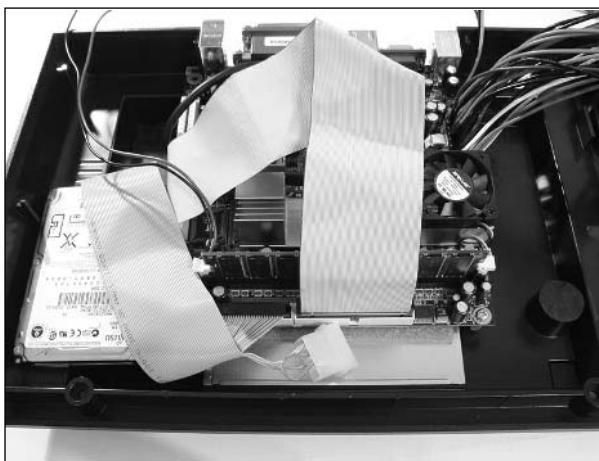
Figure 2.71 Attaching Foam Tape to the Underside of the Hard Drive



2. Remove the protective coating from the bottom side of the foam tape and mount the drive to the front left area of the bottom Atari housing. The foam tape will stick to the lip of the plastic. The IDE connector of the drive should be facing toward the back of the case.
3. Before connecting the IDE cable to the motherboard, retrieve the two-wire red and black cable that you cut off the iTuner Power Supply Module. Plug it into pins 6 and 8 of the header marked “F PANEL” on the silkscreen (shown in Figure 2.14). The header is immediately next to the hard drive. It doesn’t matter which orientation the connector goes on, since both leads will be connected to the momentary power switch on the control panel later in the hack.
4. Finally, flip the IDE cable over the hard drive and connect it to the IDE connector marked “SECONDARY” on the silkscreen (the connector closest to the drive).

Your system should now resemble the one shown in Figure 2.72.

Figure 2.72 The Hard Drive Mounted into the Atari 2600 (Front Left Corner)



The PW70 Power Supply Module

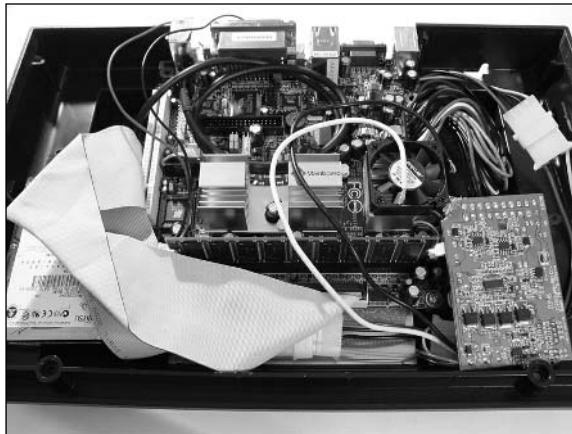
Fitting the PW70 Power Supply Module into the case is one of the trickier aspects of this hack. The ATX connector is quite high compared to the amount of free vertical space we have inside the Atari case, so finding the best location to mount the unit might take a bit of tweaking.

1. First, connect the ATX Extension Cable (which is now soldered to the motherboard) into the ATX connector of the Power Supply Module. Next, place the Power Supply Module upside down at the front right corner of the Atari housing. Feed the cables underneath the module to reach the power connectors of the CD-ROM and hard drive. The black and white wires should also be brought toward the back of the system, since they will be soldered onto the power connector on the control panel later in the hack.

2. The module should stay in its place without any mounting materials, but you might want to tack it down with a little bit of hot glue just so it doesn't move around as you continue placing components into the housing. Finally, plug the PC power connectors into the hard drive and CD-ROM.

Your system should now resemble the one shown in Figure 2.73.

Figure 2.73 The Power Supply Module Mounted into the Atari 2600 (Front Right Corner)



The USB Components

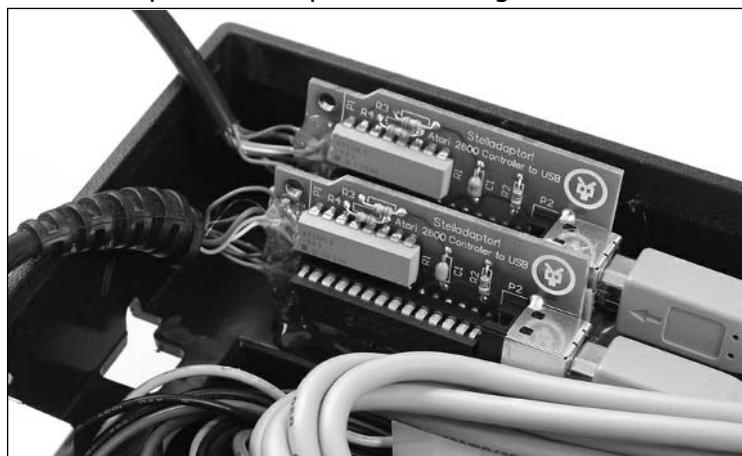
The next step is to cram all the USB components into the case. This includes the mini four-port USB hub, cordless keyboard/mouse receiver, 802.11b NIC, and two Stelladaptors. These parts will essentially fit wherever they can inside the case. Since they are all connected via standard USB cables, there will be sufficient length to place the devices in every nook and cranny within the housing.

1. First, plug in the modified USB header into the yellow connector on the motherboard marked “USB3/4” on the top silkscreen. Connect one side of the Type A male-to-female USB cable extender to the D-Link USB NIC, and connect the other side to one of the connectors of the modified USB header.
2. Next, attach a 2-inch piece of foam tape to the underside of the D-Link USB NIC. This device fits perfectly in the empty area directly underneath the hard drive. Carefully slide it underneath and press it down to secure it to the bottom housing.
3. Next, attach a 4-inch piece of foam tape to the underside of the mini four-port USB hub. Mount the unit lengthwise on the back left side of the Atari case, behind the hard drive. One end of the mini-hub actually sits on top of the hard drive and angles downward toward the back of the case. Be careful not to cover the hole on the hard drive marked with a “DO NOT COVER” warning. This is an air vent for the drive; if covered, it can cause the drive

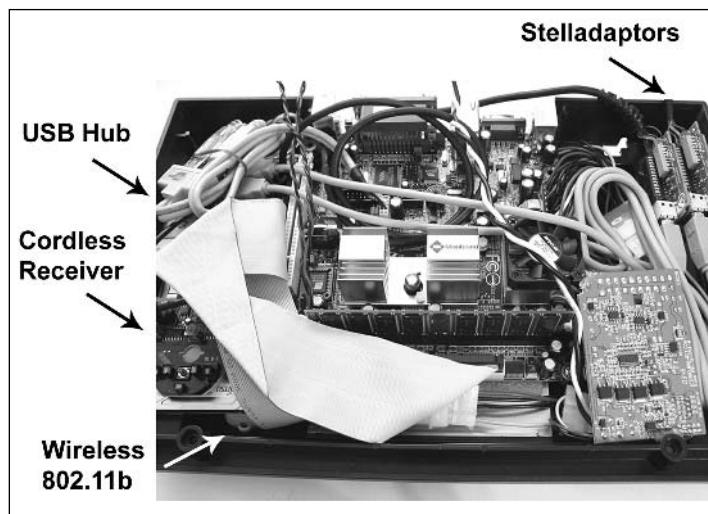
to overheat and/or fail. Connect the mini-hub connector to the other remaining connection on the modified USB header.

4. Now we will mount the cordless keyboard/mouse receiver circuit board to the top of the hard drive. Place two short lengths of foam tape across the solder side of the circuit board (the side with no components). Then, press it down on top of the hard drive, again taking care to not cover the hole on the hard drive marked with a “DO NOT COVER” warning.
5. The final additions are to mount the two Stelladaptors into the back right corner of the housing, above the tangle of wires coming from the ATX power supply connector. Both Stelladaptors can fit nicely standing on their sides. Run the USB cable from each Stelladaptor across the motherboard in front of the processor and heat sink, and connect it into the mini-hub. Position the Stelladaptors as shown in Figure 2.74, with the USB connector toward the front of the case and the nine wires toward the back. Use hot glue to fix the modules into place.

Figure 2.74 A Close-Up of Stelladaptor Positioning

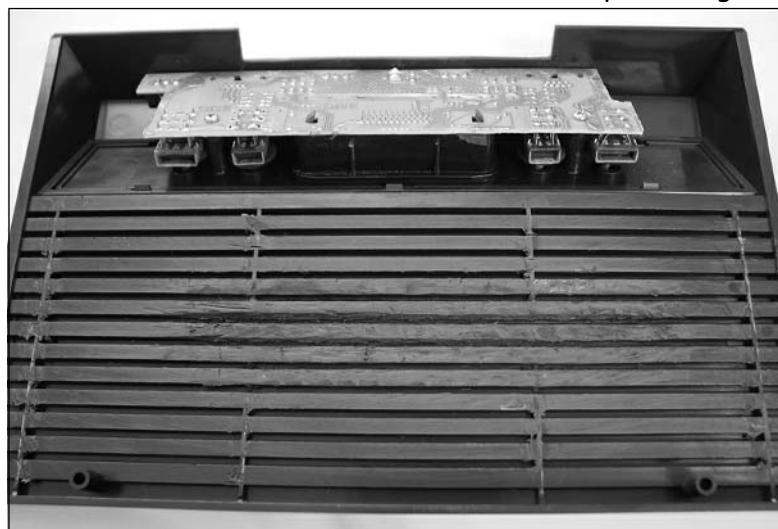


With all the USB peripherals in place, your system should now resemble the one shown in Figure 2.75.

Figure 2.75 All USB Components Mounted into the Atari 2600

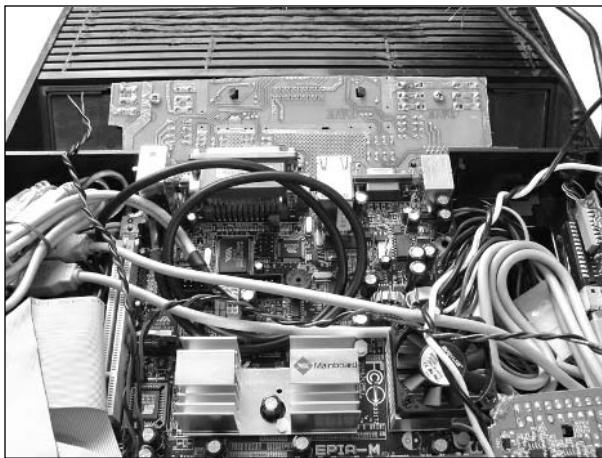
The Control Panel

Remember that control panel we spent so long preparing at the beginning of the hack? Well, it's time to mount it into the case. Before doing so, don't forget to replace the four circular pads on to the tops of each switch. With the control panel in place, use two of the screws that you set aside when you opened the case and screw them into the two holes in the control panel (see Figure 2.76).

Figure 2.76 The Control Panel Mounted to the Atari 2600 Top Housing

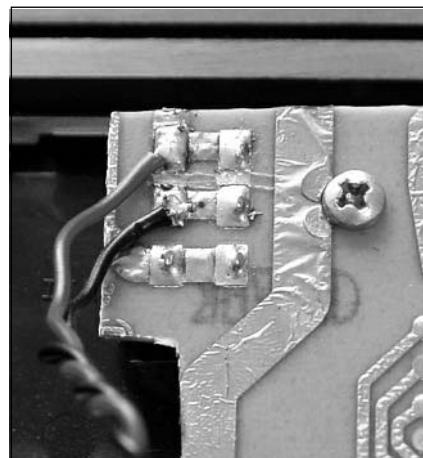
Now we need to solder all the control panel-related wires onto the control panel. The easiest way to attach the wires is to lay the top of the Atari housing (now containing the control panel) upside down, facing back to back with the bottom of the Atari housing, as shown in Figure 2.77. This way, you have easy access to both the top and bottom housings and the necessary wires and solder pads.

Figure 2.77 The Atari 2600 Housings Aligned and Ready for Connections



1. First, solder the red and black leads from the F PANEL header to the two upper left pads of the S201 momentary power switch (on the far left of the control panel, if you're looking at the solder side of the circuit board, as shown in Figure 2.78). The polarity of the connections does not matter for this switch.

Figure 2.78 Connecting Wires to the Power On/Off Switch



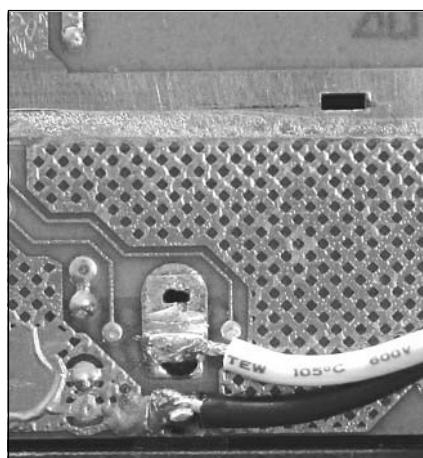
- Now, solder the two wires from the cordless keyboard/mouse extension cable to the two upper left pads of the S202 momentary switch (on the far right of the control panel, if you're looking at the solder side of the circuit board, as shown in Figure 2.79). The polarity of the connections does not matter for this switch.

Figure 2.79 Connecting Wires to the Cordless Keyboard/Mouse Connect Switch



- Next, solder the black and white wires from the Power Supply Module onto the power jack of the control panel. The black (GND) wire connects to the front pad (closest to you, as shown in Figure 2.80) and the white (+12VDC) connects to the other pad. The polarity of the connections is crucial in this step. Ensure that the black and white wires are connected to the proper pads before applying power to the system.

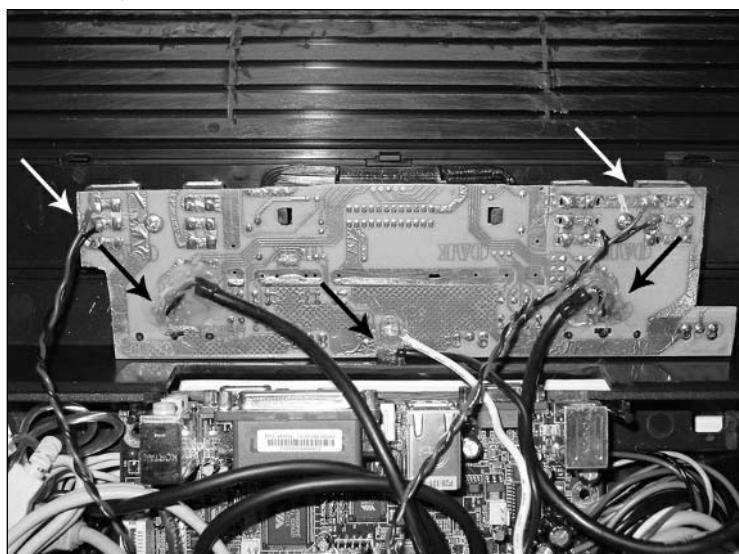
Figure 2.80 Connecting Wires to the Power Jack



4. Finally, it's time to attach the nine wires from each Stelladaptor to the DB9 connectors on the control panel. Solder pin 1 to pin 1, pin 2 to pin 2, and so on for each Stelladaptor-to-DB9 connection. Make sure that you note the proper pinout of the DB9 connectors before you solder your wires. It is easy to confuse the order and direction of the pins. If your Stelladaptors don't work properly when you've finished your hack, incorrect ordering of the wires is the most likely problem.
5. With all the wires soldered to the control panel, cover the connections with hot glue to protect and reinforce them. The hot glue will also serve as a strain relief to prevent the wires from breaking when you are closing the Atari case.

Your control panel should now have three pairs of wires and the DB9 connections soldered to it. It should resemble the system shown in Figure 2.81.

Figure 2.81 The Fully Connected Control Panel



At this point, we've made all the connections for the hack and we can button up the case!

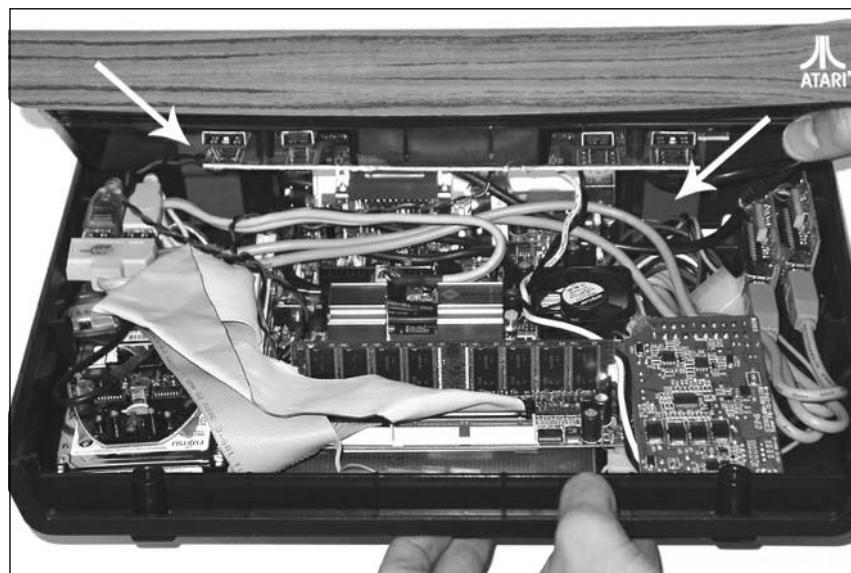
Closing It Up: Completing the Atari 2600PC Case Modification

It's *finally* time to close up the case!

1. First, make sure that all the components and cables are properly connected. Now would be a good time to test the PC's functionality again, to make sure everything works as it did before it was force-fit into the case.

2. Next, place the top housing onto the bottom housing, taking extreme care not to crush or pinch any wires or components or accidentally bend the motherboard. If the case does not close easily, take a peek inside to see what is preventing it from doing so. Chances are, it's just something simple like a wire or cable getting in the way, so you can try to move things around until the case closes nicely. If there is another problem, remove the top housing and remedy the situation. Take care not to crush the CD-ROM drive, the DRAM, or the motherboard when you are cramming the case halves together.
3. Once the case halves fit together, you will notice that the back half is still slightly open, since the original screw mounts were removed from the bottom of the case in order to fit the motherboard. So, use two short lengths of foam tape on the inside back halves of the case to hold the back shut. The foam tape will act as a hinge (see Figure 2.82). If you are feeling ambitious, you can use two short lengths of the hanger strap to create hinges, drilling them into the inside back of the case and mounting them with screws and nuts. Either method works just fine, though the hanger strap will be sturdier than the foam tape and won't deform with heat.

Figure 2.82 Using Foam Tape to Act as Hinges



4. To finish it off, screw the two front screws into the front of the case (denoted in Figure 2.7).

Congratulations! The case modification is complete! Figures 2.83, 2.84, 2.85, and 2.86 show different views of my Atari 2600PC.

Figure 2.83 The Completed Atari 2600 Case Mod



Figure 2.84 The Completed Atari 2600 Case Mod: Close-Up Showing CD-ROM Drive



Figure 2.85 The Completed Atari 2600 Case Mod: Back View



Figure 2.86 The Atari 2600PC in Action



In Conclusion...

Overall, I am pleased with the way the hack turned out. It took a lot of time and effort to come up with creative ways to fit all the gear inside the case.

The Atari 2600PC boots rapidly and has no problem running the various emulators and CyberLink's included PowerDVD player. If connected to a computer monitor, the Atari 2600PC could easily be used for traditional PC tasks, such as surfing the Web, sending and receiving e-mail, and word processing, all without noticeable lag.

The fan is quiet, though audible enough that I can confirm that the system has properly powered down after I use it, if I listen carefully. The system is connected to my home wireless network, which makes it a snap to load updated emulators and new game ROMs onto it.

When I play DVDs, which uses a fair amount of computational power, the system gets pretty hot. However, I have yet to experience any problems with overheating. Cooling features could always be retrofitted to the hack, if necessary.

My main concern is that the wireless keyboard/mouse, now that it has been modified, has a range of only about 1 foot compared to the original 6 feet it had before modification. This could be due to electrical noise or the fact that the receiver is blocked by the thick plastic Atari housing. So, I've resorted to using a wired USB keyboard and mouse at certain times, unless I'm directly in front of the system.

As a final touch, you could pick up a new-old-stock original Atari dust cover (model #CB101188-GL) from Best Electronics (www.best-electronics-ca.com). It is brown vinyl with a gold Atari logo. This way, the unit won't get too dusty when it isn't in use, and the dust cover looks cool, too.

Enjoy your new retro PC!

Resources and Other Hacks

Case Modifications on the Web

Hundreds, if not thousands, of Web sites are dedicated to case modifications and serve as showcases and galleries for hardware hackers to show off their latest creations. If you're looking for ideas or inspiration or want to hook up with other case modders, these sites are a great place to begin:

- **VIA Mini-ITX motherboard** www.mini-itx.com
- **VIA Arena** www.viaarena.com
- **The Ultimate Computer Case Mod Web site** www.thebestcasescenario.com
- **PimpRig: Smack Ya Rig Up!** www.pimprig.com
- **Lucent Rigs** www.lucentrigs.com
- **The Mod Asylum** www.modasylum.com

Stuffing PCs into Videogame System Consoles

There has always been something cool about creating a usable computer system from an old videogame console. The following list has a few of the interesting ones that other people have worked on:

- **NESPC** www.mini-itx.com/projects/nesc
- **Atari 800 ITX** www.mini-itx.com/projects/atari800
- **AnimalSNES** www.mini-itx.com/projects/animalsnes
- **PlayStation PC** www.mini-itx.com/projects/playstationpc
- **PlayStation2 PC** www.mini-itx.com/projects/playstation2pc
- **Dreamcast PC** www.mini-itx.com/projects/dreamcastpc
- **Saturn PC** www.mini-itx.com/projects/saturnpc

Creating Your Own Portable Game System

A few individuals have made great strides in creating their own portable game systems by hacking the original systems into something smaller. These miniature case modifications are a niche area, a subculture, if you will, and the final result can get you a lot of attention:

- www.benheck.com
- www.doomportables.org
- www94.pair.com/jsoper/port_2600_main.html
- <http://tripoint.org/Kevtris>

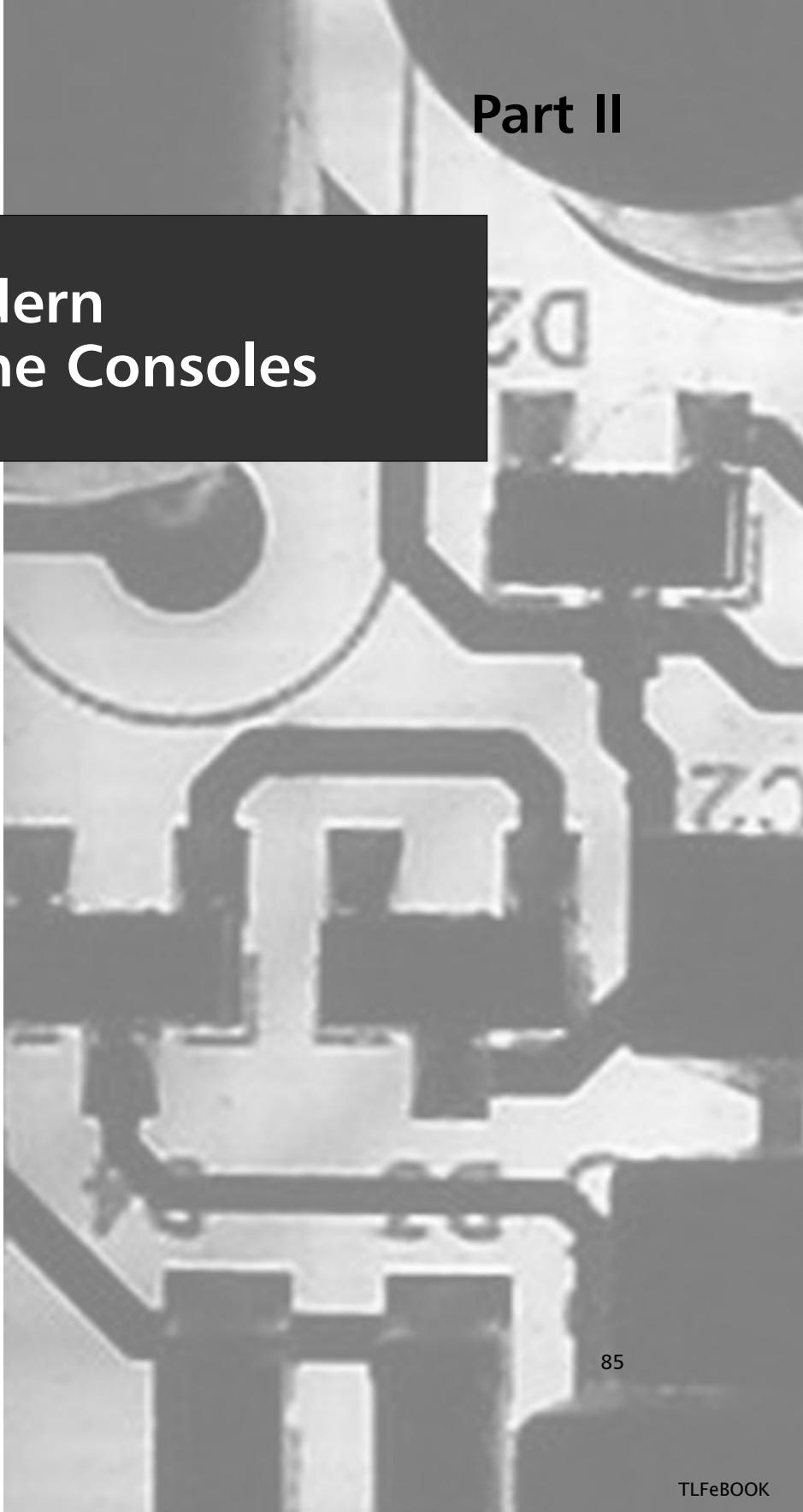
Parts and Materials

The following are good starting points to find vendors for hardware hacking materials and parts:

- **accupc.com** Low-cost U.S. distributor for VIA Technologies motherboards
- **Digi-Key Corporation** www.digikey.com
- **Tap Plastics** www.tapplastics.com
- **Case Mod Supplies** www.xoxide.com
- **Directron.com** Case modification supplies and components:
www.directron.com/mods.html
- **Best Electronics** The world leader in Atari replacement parts and accessories: www.best-electronics-ca.com

Part II

Modern Game Consoles



Chapter 3

Xbox

Topics in this Chapter:

- **Introduction**
- **Opening the Xbox**
- **Controller Hacks**
- **Xbox Networking Hacks**
- **Wireless Networking**
- **Installing a Modchip**
- **Running Linux on an Unmodified Xbox**
- **Other Hacks**
- **Homebrew Game Development**
- **Xbox Resources on the Web**

**Chapter Note: Circumventing copy protection measures (for example, by using modchips) in order to play pirated or unauthorized games may be unlawful in your country. Both U.S. and other governments have successfully prosecuted sellers of modchips and similar devices. Please check with your country's laws regarding such activities.

Introduction

In November 2001, Microsoft introduced the Xbox gaming console (see Figure 3.1). Microsoft's clear intent was for the Xbox to be in direct competition with other videogame consoles, such as Sony's PlayStation and PlayStation 2 and Nintendo's GameCube. However, unlike these other game consoles, the Xbox is not based on a specialized hardware platform designed specifically for playing games. Instead, Microsoft based the Xbox on the same hardware that made Microsoft a household name, the Personal Computer (PC). The hardware used in the Xbox is derived from the same electronic architecture as the standard "IBM-compatible" PCs.

Figure 3.1 Microsoft's Xbox

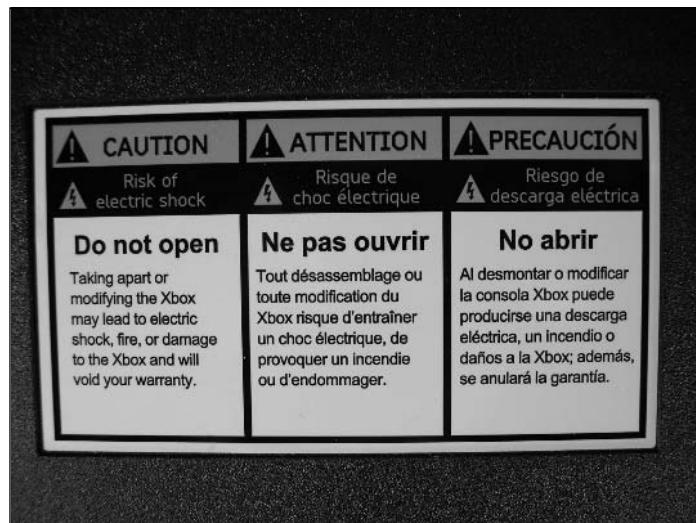


Many hardcore gamers, especially those who own different brands of game consoles, claim that since the Xbox's hardware was not designed from the ground up for gaming, it is inferior to other modern consoles. Ardent Xbox fans usually counter that this is nonsense, and that rather than reinventing the wheel, Microsoft created a superior console by using existing, tried-and-true technologies.

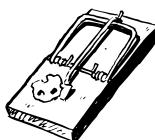
Since PC architecture is used as the basis for the Xbox, it is a very attractive option to hack, modify, and possibly run standard PC programs on. No matter what their view as to the gaming capability of the Xbox, few would argue with the fact that the PC-based architecture makes the Xbox one of the easiest game consoles on which to perform hardware hacks. People looking to hack the Xbox don't face the daunting task of learning a new hardware structure. They can eliminate a long learning curve and jump right into the modifications.

As you can see in Figure 3.2, hacking the hardware of your Xbox will, of course, void your warranty. But isn't that part of the fun?

Figure 3.2 Voiding Your Warranty: The First Step in Hardware Hacking



WARNING: HARDWARE HARM



Static electricity can cause unseen damage to electronic components. You must always be grounded before touching the inside of your Xbox, to reduce the risk of damaging these sensitive components. The easiest way to ground yourself is to buy an antistatic wrist strap and connect it to a grounded source. If you do not have a wrist strap, you can ground yourself by touching a piece of metal (such as the edge of your desk) before touching the Xbox's circuitry.

Xbox Hardware and Specifications

The Xbox platform is based on the following hardware:

- An Intel Celeron-class processor running at 733 MHz
- An nVidia GeForce 3MX Graphics Processing Unit (GPU) running at 233 MHz

- An nVidia Media Communications Processor (MCPX)
- Sixty-four MB of RAM
- An 8 or 10 GB hard disk
- DVD drive (4x speed read capability)
- A 10/100 Mbps RJ-45 Ethernet network port

Looking over this list, you'll notice that the hardware is not dramatically different from a PC circa late 2001 or early 2002. Some of the items on the list are specialized for the Xbox but have comparable parts in a PC. For example, the nVidia GeForce 3MX GPU is the Xbox counterpart of the video display adapter card found on most PCs.

Xbox Versions

Most of the hacks detailed in this chapter will work on any Xbox, but some modifications depend on a particular version of the Xbox. For example, some models of daughter boards, or modchips, that load alternate firmware are very specific as to which version of the Xbox they will run on, whereas others might be flexible enough to run on any current version. (See the “Installing a Modchip” section for more details.) The information presented in this section will help you determine which version of the Xbox you own.

As of this writing, there are seven known differences in Xbox hardware, designated as versions 1.0 through 1.6. These version numbers are not “official,” since Microsoft does not use them, but the worldwide community of Xbox hardware hackers have come up with the scheme to more easily keep track of system variances. The Web page at www.xbox-scene.com/versions.php will step you through a series of online prompts to help you determine which version of the Xbox you own.

Table 3.1 shows some of the hardware differences in the various versions. The differences are all internal to the console, so you'll have to open your Xbox to investigate. Some differences are very easy to see, such as the Universal Serial Bus (USB) daughterboard in version 1.0. Other changes are subtler.

Table 3.1 Xbox Versions and Features

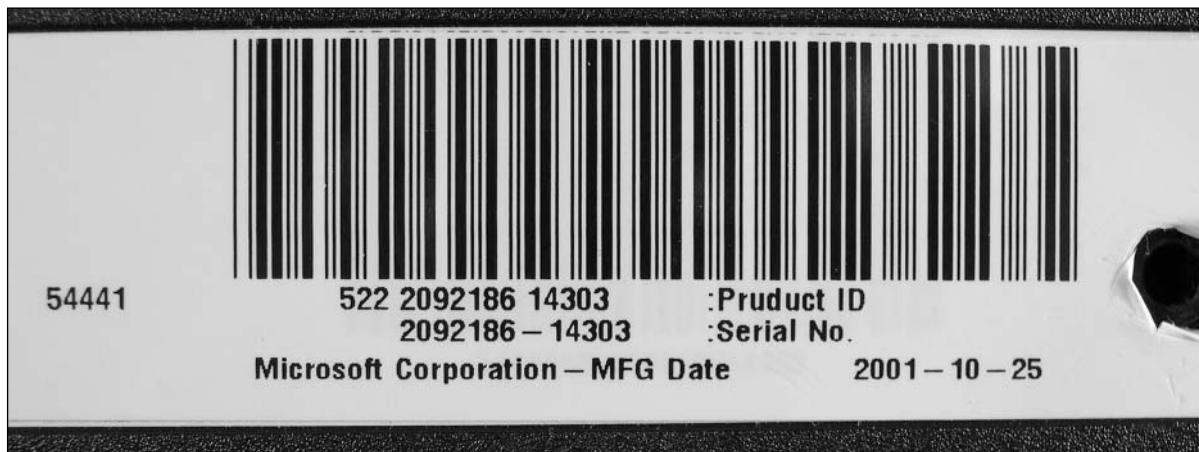
Feature/Capability	1.0	1.1	1.2	1.3	1.4	1.5	1.6
GPU fan	X						
USB daughterboard	X						
Single-row power connector	X	X					
Double-row power connector			X	X	X	X	X
Video chip labeled “Conexant”	X	X	X	X			
PAL Special Edition translucent green Xbox (not the NTSC Special Edition Halo Xbox)				X			
Video chip labeled “Focus”					X	X	

Continued

Table 3.1 Xbox Versions and Features

Feature/Capability	1.0	1.1	1.2	1.3	1.4	1.5	1.6
3.3V and ground are connected on the LPC bus					X		
3.3V and ground are disconnected on the LPC bus						X	
Video chip labeled “xcaliber” with Xbox logo							X
Halo Special Edition translucent green Xbox					X		X

The Web page at <http://xbox-linux.sourceforge.net/docs/versionsfinding.html> is an excellent resource to determine the exact version of your Xbox. To use the page, look up the Manufacturing Date (MFG. DATE), the Serial Number (SERIAL NO.), and the Product Identification Number (PRODUCT ID) printed on a silver, bar-coded sticker on the bottom of the Xbox. The sticker is in the upper-right center as you face the front panel of the Xbox. The label is shown in Figure 3.3.

Figure 3.3 The Product ID, Serial Number, and Manufacturing Date Label

The format of the Manufacturing Date is YYYY-MM-DD, where:

- **YYYY** Year
- **MM** Month
- **DD** Day

The format of the Serial Number is PN~~NNNNNN~~NYWWCC, where:

- **P** is the production line number.
- **NNNNNN** is a sequential number within a week.
- **Y** the last digit of the production year.
- **WW** is the number of the week (01 through 52).
- **CC** is the Country Code (see Table 3.2).

Table 3.2 Xbox Country Codes and Versions

Code	Country	1.0	1.1	1.2	1.3	1.4	1.5	1.6
02	Mexico (Prior to November 2002)	X						
02	Mexico (November 2002 and later)		X					
03	Hungary		X					
05	China		X	X	X	X	X	X

Opening the Xbox

To perform most of the hacks in this chapter, you will need to open the Xbox console. This section covers the details of how to do so.



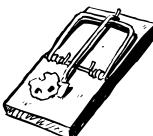
Preparing for the Hack

The tools required for this hack are:

- A Torx T-20 screwdriver
- A Torx T-10 screwdriver
- An X-ACTO knife or hobby blade

Performing the Hack

WARNING: PERSONAL INJURY



Before opening the Xbox to perform any modifications, make sure that the Xbox is not plugged in or powered on. Some components in the Xbox, particularly the power supply circuitry, can store charges that can be extremely dangerous if improperly handled. Be cautious when working around the power supply.

Perform the following:

1. To open the Xbox case, you first have to access the six screws holding the case to the chassis. Four of the screws are underneath the four rubber feet, and two are located under the labels on the bottom of the Xbox (see Figure 3.4). The two screws under the labels can be felt as soft spots on the label. Neat freaks may trim the label with a hobby knife, whereas more aggressive people can simply punch through the label with the screwdriver tip.

The rubber feet are held in place by an adhesive. Carefully prying up the rubber foot from the outside edge will expose the screw head without the need to completely remove the foot. Use a hobby knife to carefully do the prying. If you do a lot of modifications to your Xbox, sooner or later the adhesive will lose its holding power, and the feet will fall off. You can purchase generic rubber feet from many stores, such as Radio Shack, although these will typically be round and not oval shapes like the feet supplied with the Xbox. If you would prefer to keep the original feet, they can be repaired using double-sided tape. Simply apply the double-sided tape to the foot and trim the tape to fit the socket where the foot sits.

2. Unscrew the six screws using the Torx T-20 driver.
3. Once the screws have been removed, gently pull the cover off the Xbox. If the cover sticks or doesn't want to come off, lightly rap it with your knuckles on all four sides. This should loosen the cover enough to allow it to be lifted off.

Figure 3.4 The Six Screws Are Hidden Underneath the Labels and Rubber Feet



4. Once the cover has been removed, set it aside. The Xbox's DVD and hard drive are now visible. Disconnect the power cables and other connectors going to the drives.
5. The DVD and hard drive are held in two drive trays. Three screws retain the drive trays (see Figure 3.5). Two are located between the DVD and the hard drive; the other is located to the left of the DVD. Please note that the IDE cable that runs between the hard drive, DVD, and motherboard normally covers one of the screws and must be moved aside so that you can access the screw. In Figure 3.5, the cable has been pushed aside to show the screw's exact location.

When the drive trays and the drives are removed, the Xbox's motherboard and power supply board are plainly visible, as shown in Figure 3.6.

Figure 3.5 The Xbox DVD and Hard Drive

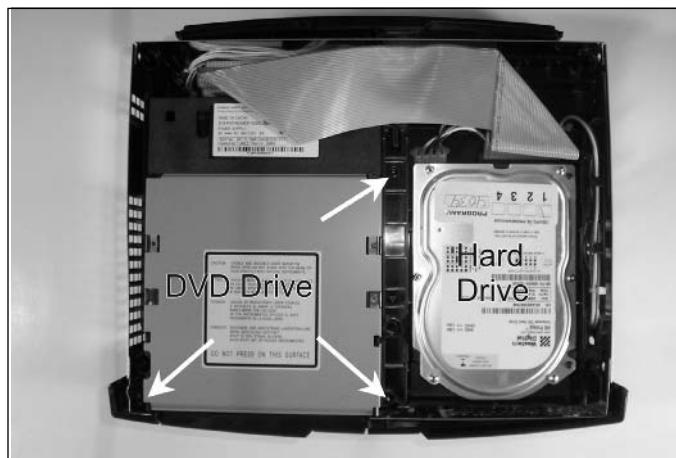


Figure 3.6 DVD and Hard Drive Successfully Removed



6. Remove the front panel by prying the three snap locks. One snap lock is located on each side next to the metal Radio Frequency Interference (RFI) shield, and the third is located in the center of the Xbox. The easiest way to open the snap locks is to pry them with a screwdriver. Do one side at a time, moving from top to bottom. When both sides are open, pry the third center lock. Be careful not to pry too hard, since it is possible to crack or break the panel.

That's it! Your Xbox should be completely open, and you should have access to all its internal circuitry.

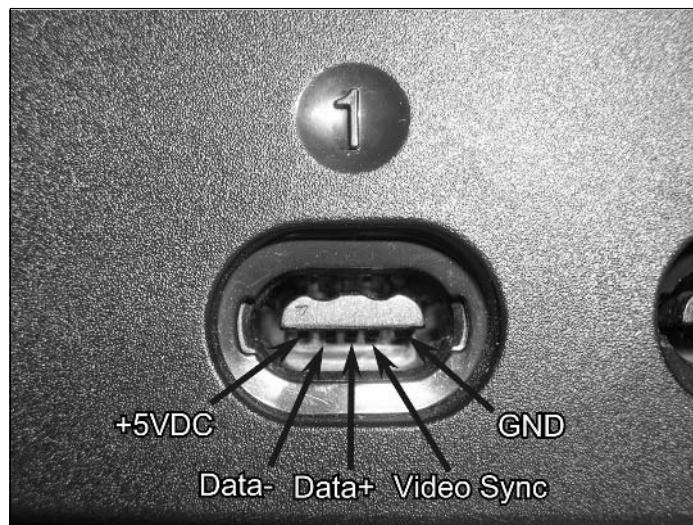
Controller Hacks

The standard game controller that comes with the Xbox, commonly referred to as the XBC on many Web sites, is a dual-handle grip design with the following components:

- Two analog thumbsticks
- A D-Pad four-way digital controller
- Four primary function buttons in a diamond placement: X (blue), Y (yellow), B (red), and A (green)
- Four secondary function buttons: White, Black, Start, and Back
- Two analog triggers, each one located on the front of the two grips

The controller uses a modified USB protocol for communications with the Xbox. By using this known standard, Microsoft made communications between the Xbox and external devices relatively easy to implement. At the front of the controller are two expansion ports for plugging in additional devices such as memory cards and communications modules, both of which are also USB-based.

A standard USB interface consists of four wires: two lines for data (Data+ and Data-) and two lines for power (+5VDC and ground). The modified Xbox USB interface carries an additional Video Sync signal. The Video Sync is designed for use with controllers that need to detect a screen position on a standard raster scan television, such as a light gun. Looking into the controller port on the Xbox, you should see five contacts across the bottom half of the port. From left to right, the pins are +5VDC, Data-, Data+, Video Sync, and GND (see Figure 3.7).

Figure 3.7 The Xbox Controller Port

Controller Versions

There are also two main versions of the standard Xbox controller: the original, or *large*, controller and the smaller S controller.

The S controller differs in two ways from the original. First, it is significantly smaller and therefore many people consider it more comfortable to use. Second, to accommodate the controller's reduced size, the button placement differs slightly. On the original controller, the secondary Black and White buttons are above the X-Y-B-A buttons, and the Start and Back buttons were located between the two thumbsticks. On the S controller, the Black and White secondary buttons are below the main X-Y-B-A buttons, and the Start and Back buttons are located toward the outside of the left grip. This change in the placement can cause some problems for players who are used to the other controller.

Functionally, the original controller and the S version are the same device. The Xbox doesn't care which type of controller you use, and the two models can be swapped interchangeably. Most of the time, the only real choice is which one fits most comfortably in your hand.

The two controllers are shown side by side in Figure 3.8. The S controller has also been released in several submodels with different colored housings, such as translucent green and translucent blue.

Figure 3.8 The S Controller (Left) and the Original Xbox Controller (Right)



A variety of third-party manufacturers, including Mad Catz and NYKO, also make controllers for the Xbox. The designs of these controllers range from simple clones of the stock controllers to wireless controllers and controllers that resemble futuristic rifles and pistols. If you don't like the controllers supplied by Microsoft, you have many other options. Most retail stores that cater to console gamers, such as GameStop and EB Games, carry a variety of controllers in different brands and designs.

Getting Inside Your Controller

This hack will guide you through the fairly easy process of opening an Xbox controller.



Preparing for the Hack

The only tools required for this hack are a Phillips-head screwdriver and an Xbox controller.

Performing the Hack

Perform the following:

1. Invert the controller and, using your Phillips-head screwdriver, remove the seven screws from the back of the controller (see Figure 3.9). Note that on the S controller, one screw is hidden behind the main label.

Figure 3.9 The Bottom of the S Controller



2. Leaving the controller inverted, pull the bottom half of the housing away from the top.
3. Once the two halves are separated, leave the top face down and place the bottom half face up (see Figure 3.10). Be careful to keep the buttons and thumbstick controls in place in the top half of the housing.

Figure 3.10 Disassembling the Controller



That's it! You are now ready to proceed with the additional hacks outlined in this section.

Illuminating the Controller Buttons with LEDs

This hack brings a cool visual element to the standard Xbox controller by adding light-emitting diodes (LEDs) to the controller's buttons. This hack will work on either the original controller or the newer S controller.



Preparing for the Hack

The components required for this hack are:

- One 100 ohm, 5%, 1/4W resistor (Radio Shack #271-1311).
- Four LEDs, 1.7V to 5V forward voltage @ 20mA, size T-1 (3mm) or T-1-3/4 (5mm). You can either use white LEDs (Radio Shack part #276-320) for all four buttons or use specific colored LEDs based on the plastic color of the button (red, blue, green, and yellow). Note that different color LEDs have different characteristics and may require a change in resistor value (see the “Under the Hood: How the Hack Works” section for more details).

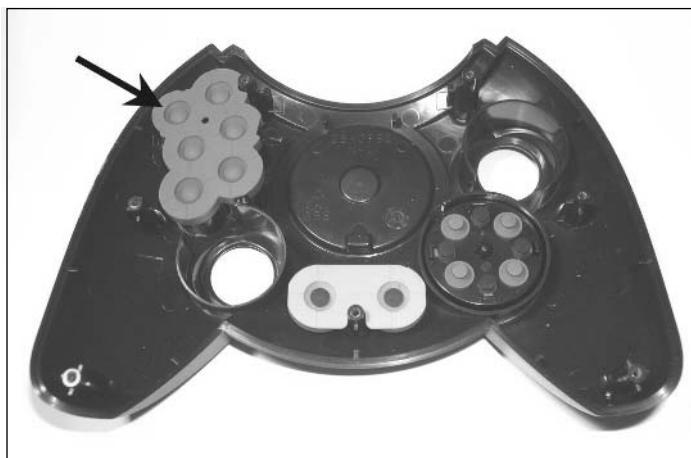
The tools required for this hack are:

- Phillips screwdriver, size #1
- Soldering iron and solder
- Dremel tool or drill
- Wire, 4 feet total (approximately 2 feet each of two colors, preferably red and black)
- Electrical tape or a heat gun and heat-shrink tubing
- X-ACTO knife or hobby blade

Performing the Hack

Perform the following:

1. Open your controller as described in the “Getting Inside Your Controller” section earlier in this chapter. We will be working with the top half of the controller housing, which contains the plastic button pieces.
2. Remove the rubber conductive pad (see Figure 3.11) to gain access to the plastic button pieces.

Figure 3.11 Removing the Conductive Pad

3. You can now remove each of the four plastic button pieces: X is blue, Y is yellow, A is green, and B is red. Figure 3.12 shows the removed plastic button and LED we'll be putting in.

Figure 3.12 The Red "B" Button and Red LED

For the LED to fit into the plastic button housing, we'll need to drill out some space.

4. Carefully drill into the white plastic on the interior of the button. Choose a drill that is closely matched to the diameter of the LED's lens. For a standard T-1-3/4, a 3/16-inch drill bit will be a good fit. Compare Figures 3.13 and 3.14 for before and after views of the area to be drilled out. Work slowly and carefully, making sure you don't drill right through the button. If necessary, trim the interior plastic so that the LED has a perfectly snug fit.

5. On the side of the button, cut a groove so that the LED leads can exit the button. On these buttons, the groove was placed above the little index tab that orients each button. This groove will allow us to wire the LEDs in the next step.

Figure 3.13 Button Underside Before Drilling

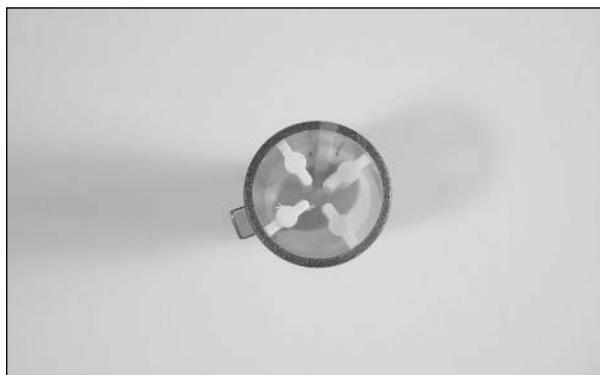
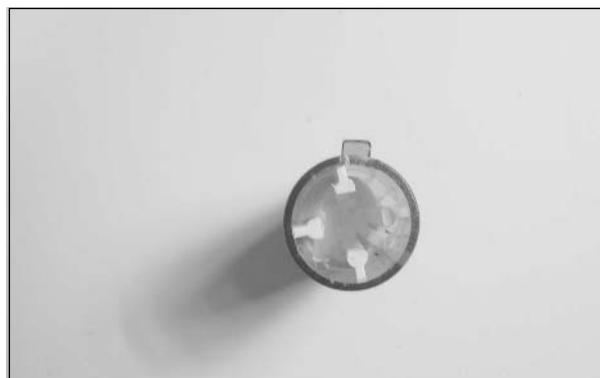
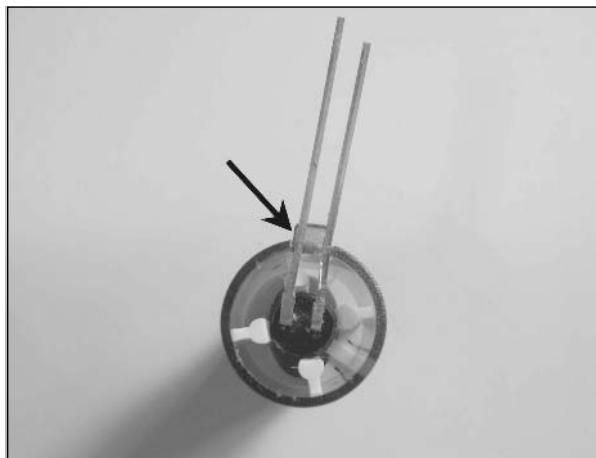


Figure 3.14 Button Underside After Drilling



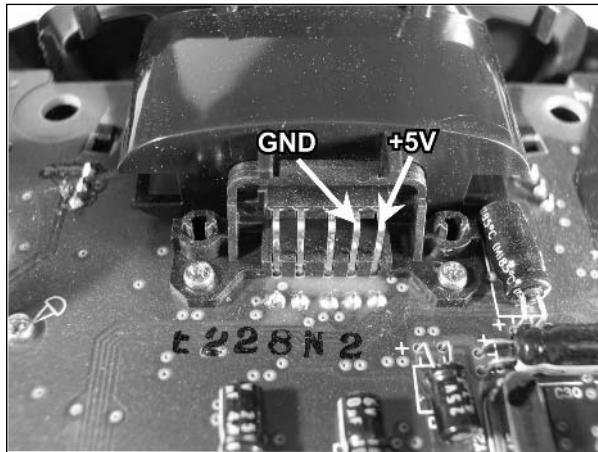
6. Bend the leads of the LED to about 90° (see Figure 3.15). This will allow the leads to exit the side of the button so that wires can be attached.
7. Align the bent leads of the LED with the groove you cut in Step 5, and insert the body of the LED into the button. If the drilled-out portion is the correct size, the button should slide in without too much effort. If it does not, do not force it. Remove the LED and trim the white plastic with the knife. When inserted into the body of the button, the leads should be sticking out through the groove, as shown in Figure 3.16.

Figure 3.15 LED Leads Bent 90 Degrees**Figure 3.16 The LED Inserted into the Button**

8. When the LED is installed into the plastic button housing, the leads of the LED should stick out slightly above the existing posts. Trim the leads for the LED so that they stick out above the button by approximately 1/8 inch.
9. Now replace each button into its original location on the controller's top housing.
10. Repeat Steps 1 through 9 for each of the four buttons.
11. Now it's time to attach wires to the LEDs. First, cut your wire into eight pieces of six-inch lengths. Two wires, one red and one black, will be used per button.
12. Solder one end of the black wire to the cathode of the LED (the side closest to the flat edge of the LED).

13. Solder one end of the red wire to the anode of the LED (the remaining free pin on the LED).
14. Solder the other end of the red wire to one end of the 100 ohm resistor. When installing each wire, you will need to make sure that the buttons have enough wire so that they are still free to move up and down. You might have to carefully remove some plastic from the button's shaft to allow free movement.
15. Now solder the other end of the black wire to the ground connection on the controller (see Figure 3.17).
16. Finally, solder the remaining end of the resistor to the +5V connection on the controller (see Figure 3.17).

Figure 3.17 The Locations for +5V and GND on the Controller



17. Repeat Steps 11 through 16 for each of the four LEDs. When all of your wiring is complete, you will have a rat's nest of wires inside the controller.
18. Before going any further, plug in the modified controller into the Xbox and turn on the system. The LEDs should all illuminate. The buttons should have a soft glow behind them. If one or more LEDs don't light up, see the following "Testing and Troubleshooting" section.
19. Now you can reassemble your controller. This is just a matter of tucking all the new wires into the housing and screwing the case back together.

Congratulations! Enjoy your newly lighted buttons!

Under the Hood: How the Hack Works

The "Blue Power LED Modification" hack in the Nintendo NES chapter and the "Blue LED Modification" hack in the Atari 7800 chapter both feature details of Ohm's Law and why the partic-

ular LED and resistor values were chosen. In our hack, the 5 volts used to power the LEDs is obtained through the Xbox controller's USB lines, and we use a 100 ohm resistor to limit the current flow through each.

Testing and Troubleshooting

The most common problem when installing LEDs into a system is incorrect polarity. Knowing which lead is the anode and which is the cathode is easy when the leads are their original length and you can see the full LED. Once they are installed into the button housings, it becomes more difficult.

If one or more of the LEDs light (but not all of them), the polarity to the nonlighting LED is likely backward. If none of the LEDs light, it's possible that the polarity is backward for all the LEDs. Either way, simply unsolder the two connections to each of the troublesome LEDs and reverse their locations.

Optional Hack: Illuminating the Controller Logo

Since we already have a voltage source in the controller, an optional hack is to place additional LEDs into the controller to illuminate the plastic Xbox logo. The actual plastic piece containing the logo will have to be removed from the controller and the green paint sanded off the back of the logo (to allow light to pass through the plastic). Behind the logo, the plastic housing of the controller will have to be opened by cutting or grinding to allow the LEDs to fit.

Adding a Remote Reset Switch

In a perfect world, games wouldn't freeze or get stuck in some infinite loop. Unfortunately, the world isn't perfect, and games do occasionally go crazy, despite the best efforts of programmers and developers. When this happens on an Xbox, the user has to power off the Xbox or eject the DVD to reset the game. Although this isn't a big deal if the Xbox is within reach of the user, it can be troublesome if the Xbox is located several feet away and you don't want to lose your perfect position on the couch.

Adding a Remote Reset Switch to the Xbox Controller

This hack places a reset switch in a convenient location: right on the Xbox controller! This is done by rerouting the connections from the original Xbox reset switch to a new switch mounted on your controller.

Two cautions about this hack: First, the hack is specific for a particular controller port on the Xbox. Plugging in the hacked controller to an unmodified Xbox port might cause problems. Second, this hack removes the ability to use a light gun or other type of controller that uses the video sync signal on the modified controller port.



Preparing for the Hack

The only component required for this hack is a push-button SPST switch (momentary, miniature size) Radio Shack part #275-1547. The tools required for his hack are:

- A Phillips screwdriver, size #1
- A soldering iron and solder
- A Dremel tool or drill
- Twenty or 22 gauge wire, three pieces, approximately 6 to 8 inches long
- Electrical tape or a heat gun and heat-shrink tubing

Performing the Hack

This hack consists of two parts: adding the switch to the controller and modifying the video sync wire on one of the Xbox's controller ports.

Perform the following:

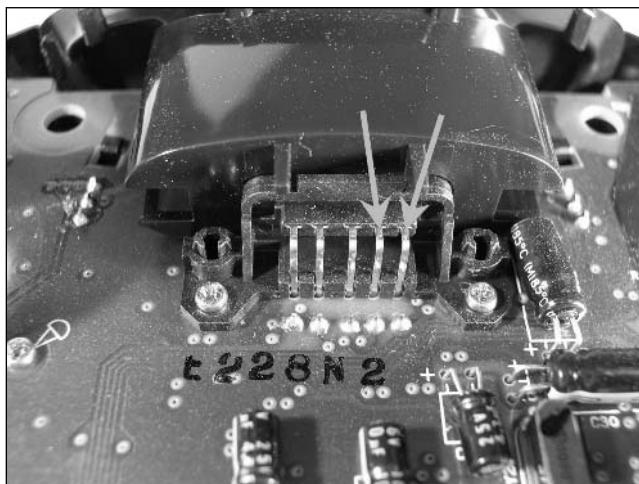
1. Open the controller as described in the previous “Getting Inside Your Controller” section.
 2. Select a location for the switch. This is somewhat critical because it must fulfill three requirements:
 - The switch should fit within the controller and not interfere with the other parts.
 - It must be easy to reach when you want to reset a game.
 - It must not be in a location where you’ll reset the Xbox inadvertently.
- One of the preferred locations is the underside of the controller, centered between the two handgrips (see Figure 3.18). However, the actual location will depend to a great extent on the size of the switch you’ve chosen.

Figure 3.18 Preferred Location for the Reset Button



3. Drill a hole for the switch and mount it within the controller. The switch should have instructions showing the size needed for the mounting hole. Most switches utilize a nut on a threaded shaft to secure the switch.
4. Take two pieces of wire and solder one end to the each lead of the switch.
5. Now solder the other ends of the wires to pin locations 1 and 2 on the controller's circuit board (see Figure 3.19). These locations are just behind the socket for the Memory Card or Xbox Live Communicator. The polarity of these wires does not matter, since we are simply connecting a momentary switch.
6. Now you can reassemble your controller by following the disassembly steps in reverse.

Figure 3.19 The Connections for the Reset Switch



With the reset switch successfully added to the controller, we can move on to modifying the Xbox console itself.

7. Open the Xbox case and remove the front panel as described in the previous “Opening the Xbox” section.
8. Remove the front control panel circuit board.
9. Choose the port you want to use to allow the modified controller. Five wires come from each controller port. Locate the yellow wire coming from the desired controller port and cut it as close to the Xbox motherboard as possible.
10. Solder one end of your remaining piece of wire to the yellow wire that you just cut. Be sure to insulate this connection with electrical tape or with a heat gun and heat-shrink tubing.
11. Solder the other end of wire to the second connection from the bottom on the front control panel circuit board.

12. Now reassemble the Xbox by following the disassembly steps in reverse.
- That's it! The hack is complete, and you should have a nice, convenient method of resetting your Xbox from the comfort of your favorite chair.

Testing and Troubleshooting

To test the hack, plug the controller into the modified port of your Xbox and power on the console. Once the Xbox is running, pressing the new reset switch on your controller should reset the Xbox. If the reset switch does not work, first make sure that you have plugged the controller into the modified port of your system. If it still doesn't work, reopen the Xbox and the controller and carefully double-check your wiring and solder connections.

Adding a Remote Reset Switch to the Xbox Controller Memory Card or Xbox Live Communicator

This hack moves the reset switch from the Xbox console onto an Xbox Memory Card (see Figure 3.20) or Xbox Live Communicator. We will be using an Xbox Memory Card to demonstrate this hack, though the hack will also work fine with an Xbox Live Communicator.

Figure 3.20 The Xbox Memory Card



As with the previous “Adding a Remote Reset Switch to the Xbox Controller” hack, this hack is specific to a particular controller port on the Xbox. Plugging in the hacked controller to an unmodified Xbox port may cause problems. Second, this hack removes the ability to use a light gun or other type of controller that uses the video sync signal on the modified controller port.



Preparing for the Hack

The only component required for this hack is a push-button SPST switch (momentary, micro-miniature size, PCB mount), Digi-Key part #CKN9017-ND. The tools required for his hack are:

- A Phillips screwdriver, size #1
- A soldering iron and solder
- A Dremel tool or drill
- Twenty or 22 gauge wire, three pieces, approximately 6 to 8 inches long
- Electrical tape or a heat gun and heat-shrink tubing
- Hot glue or epoxy
- A multimeter (optional)

Performing the Hack

This hack consists of two parts: adding the switch to the Xbox Memory Card and modifying the video sync wire on one of the Xbox's controller ports.

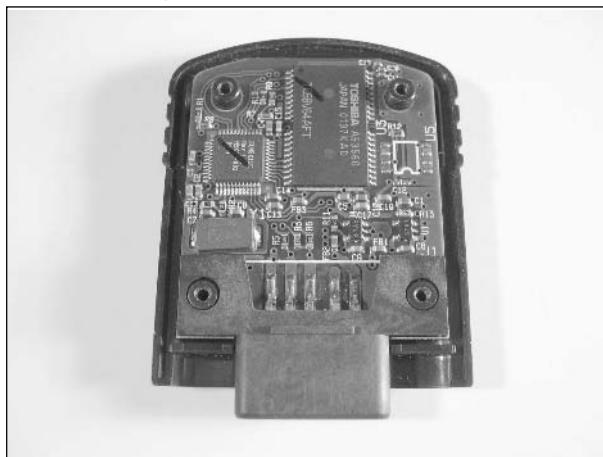
Perform the following:

1. Open the Xbox Memory Card or Xbox Live Communicator by unscrewing the four screws on the back (see Figure 3.21).

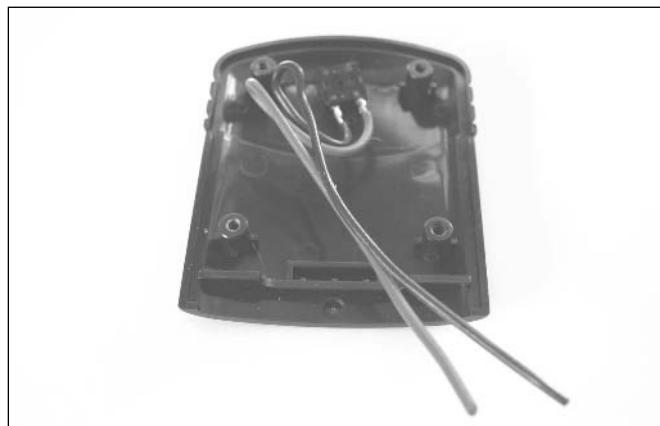
Figure 3.21 The Four Screws on the Bottom of the Xbox Memory Card



2. Remove the top half of the plastic housing and set it aside. It should come off without any trouble and should resemble the image in Figure 3.22.

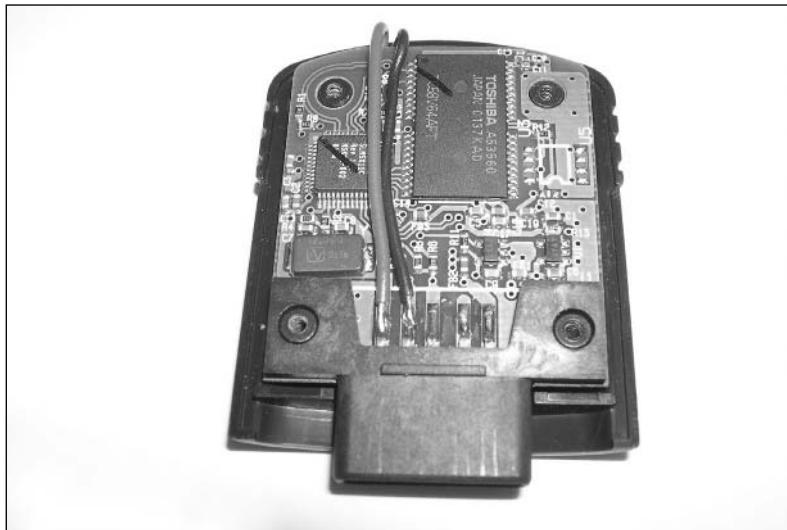
Figure 3.22 The Xbox Memory Card Circuit Board

3. Select a location for the switch. This is somewhat critical because it must fulfill three requirements:
 - The switch should fit within the Xbox Memory Card or Xbox Live Communicator and not interfere with any internal components.
 - It must be easy to reach when you want to reset a game.
 - It must not be in a location where you'll reset the Xbox inadvertently.
The actual location you choose will most likely depend on the size of the switch you are using. A suitable location is shown in Figure 3.23.
4. Drill a hole for the switch and mount it within the Xbox Memory Card or Xbox Live Communicator case. You might have to use hot glue or epoxy to secure the switch into place.

Figure 3.23 The Switch Mounted in the Memory Card Case

5. Take two pieces of wire and solder one end to the each lead of the switch. Some PCB-mount micro-miniature switches have four contacts, with two on each side connected electrically. Check the data sheet that came with your particular switch if you are unsure of which contacts to solder to.
6. Now solder the other ends of the wires to pin locations 1 and 2 on the Xbox Memory Card's circuit board (see Figure 3.24). Even though an Xbox Memory Card is shown, the connections are identical on the Xbox Live Communicator. The polarity of these wires does not matter; we are simply connecting a momentary switch.

Figure 3.24 The Xbox Memory Module Interior View



7. Reassemble the Xbox Memory Card or Xbox Live Communicator by simply placing the two case halves back together and replacing the screws.
8. The final step is to modify the interior wiring of one of the Xbox's controller ports. Follow Steps 7 through 12 as described in the previous “Adding a Remote Reset Switch to the Xbox Controller” hack.

Testing and Troubleshooting

To test the hack, insert the modified Xbox Memory Card or Xbox Live Communicator into the controller, plug the controller into the modified port of your Xbox, and power on the console. Once the Xbox is running, pressing the new reset switch on the card should reset the Xbox. If the reset switch does not work, first make sure that you have plugged the controller into the modified port of your system and that the card is seated properly in the controller. If it still doesn't work, reopen the Xbox and the card and carefully double-check your wiring and solder connections.

Adding an Xbox Live Communicator to a Wireless Controller

Wireless Xbox controllers such as the Mad Catz Lynx (see Figure 3.25) are great additions to the Xbox gaming experience, since they remove your physical link to the Xbox. However, wireless controllers lack the expansion slots typically located on the front of standard, wired Xbox controllers. The wireless base unit (receiver) that is plugged into the Xbox's controller port usually has the expansion slots, so any memory cards you use must be located close to the Xbox.

Figure 3.25 The Mad Catz Lynx Wireless Controller



Probably the biggest disadvantage to wireless controllers is that they can't use the Xbox Live Communicator without running a wire from the headset to the base unit. But if you're running a wire from the base unit to your headset, what's the point of having a wireless controller?

To totally cut the cord, you will need a wireless headset for use with your wireless controller. This hack will guide you through the steps to create one.



Preparing for the Hack

This hack is mainly a matter of gathering and assembling the right components. The materials you'll need are:

- Xbox Wireless Controller; a good choice is the Mad Catz Lynx, available for around \$30 at many retail stores such as EB Games, GameStop, CompUSA, and KB Toys.
- Xbox Live Communicator.
- A wireless headset. A good choice is the Jabra FreeSpeak model BT200 wireless headset with Bluetooth adapter (see Figure 3.26). The BT200 costs approximately \$130 and is available in many retail stores such as Radio Shack or Best Buy. Make sure that the headset you

choose has a 2.5mm micro-miniature plug on the base section, which will fit into the jack on the Xbox Live Communicator.

Figure 3.26 The Jabra Wireless Headset, Xbox Live Communicator, and Mad Catz Lynx Base



Performing the Hack

To assemble the components, perform the following:

1. Plug the Xbox Live Communicator into the Wireless Controller's base.
2. Plug the Xbox Wireless Controller base into the Xbox.
3. Plug the wireless headset jack into the Xbox Live Communicator.
4. Turn on the Xbox and the wireless headset.
5. Adjust the volume controls on the Xbox Live Communicator and the headset to your desired volume.

That's it! You now have completely cut the cables connecting you to the Xbox, so you can roam freely while you play your Xbox Live Communicator-supported games.

Xbox Networking Hacks

Many Xbox games are designed to have multiple players compete with each other, whether the players are sitting side by side or thousands of miles away connected by a network. It can be a blast to set up several Xboxes on a Local Area Network (LAN) in the same room and have a Halo Deathmatch or Crimson Skies dogfight with some of your best friends.

To get the most fun out of the Xbox's network, you have to hack it. In this section, we'll explore a couple of exciting things you can do to the network.

Establishing a Network Link Using Standard Networking

The Xbox comes equipped with a built-in standard 10/100 Ethernet network port. This port can function on 10 megabit per second (Mbps) or 100 Mbps networks. A variety of networking hubs and cables are available for purchase that will allow four or more Xboxes to be linked together. The Linksys model EFAH05W (www.linksys.com) is a good example of an inexpensive hub, available in many retail outlets such as Radio Shack or Staples, at a price of about \$40. Most come with easy-to-follow instructions, so even if you've never set up a network before, it becomes a relatively easy task.

NEED TO KNOW... XBOX NETWORKING KITS



A number of networking kits are marketed as "Xbox Compatible!" These kits usually contain an inexpensive hub and two to four Ethernet cables. Despite the "Xbox Compatible!" label, these kits are nothing but standard 10/100BaseT Ethernet components. Since the Xbox uses standard Ethernet protocols for networking, it doesn't take much to be compatible. The only thing that is usually unique about these kits is that they sport the Xbox colors of green and black.

The bottom line is that you don't need an "Xbox Compatible!" network kit to set up an Xbox home network. Standard Ethernet hubs or switches with standard Ethernet cables will work just fine and will most likely cost less, too.

Performing the Hack

To actually make a link and play a game between machines, perform the following:

1. Find a game that supports network gaming. Usually this is denoted on the game package as supporting "System Link." There are usually several numbers, such as "2-4" or "2-16," to indicate the number of Xboxes that can be linked together for that particular game. The game must be loaded onto each Xbox that is playing (this will require purchasing multiple copies of the game). Halo and Crimson Skies are a few examples of compatible games.
2. Once you've selected a game, it's time to set up the network. Connect all the components together and apply power to everything. If this is a network with a central hub or switch, you will need to run an Ethernet cable to each Xbox from the hub. If you are running the games on just two Xboxes, you can use a crossover cable to simply connect the two consoles together. (See the following "Creating Your Own Crossover Cable" hack for more details.)
3. Start the game on each system. After doing so, you should see a Multiplayer menu with a System Link option (see Figure 3.27).

Figure 3.27 Multiplayer System Link on *Halo*

4. Select the **System Link** option. This will cause the game to seek out other waiting games via your Xbox network (see Figure 3.28). Most games will time out after a minute or two if they don't connect to another game, so you will have to place all the games into the System Link mode quickly.

Figure 3.28 Searching for Networked Xboxes

Testing and Troubleshooting

If you cannot make a link between two Xboxes, the first step is to check that you have a network connection at the lowest level—physically. Check to see if the cables are plugged in firmly at both ends. If you are connecting two Xboxes directly using a crossover cable, check the connection at each Xbox. If you are using a hub or switch with straight-through cables, check each connection going from the hub to each Xbox on the network. It is possible that the cables are plugged in partially but are not making a connection. Ensure that the cables are actually plugged in by unplugging them and then firmly reseating the plugs in the jacks. You should hear a slight click as the connector securely locks into place inside the network jack.

If that doesn't work, make sure that all your equipment is powered on, including your hub and/or switch. Since hubs and switches are active devices, they need to be powered at all times. Lack of power is one of those things that is easy to overlook and can cause you to smack your forehead when you realize that your hours of troubleshooting were all due to the fact that you didn't press the On button.

If the power is on, check the cable lights at both ends. The Connection LED on the back of the Xbox should be illuminated and not blinking. If the LED is off or intermittently flashing, there is a problem with the cable or the Xbox's Ethernet port. Most often the problem is a faulty cable; try to swap it with a known working cable. If the new cable works, the original one is bad and you can discard it. If the new cable doesn't work, the problem lies elsewhere. Rarely, the Ethernet port will die or have some other unknown problem. If this is the case, the Xbox might have to be sent back to Microsoft for repair.

Once the Connection LED is lit on all the Xboxes, the next step is to check for network activity. When the Xboxes are connected and attempting to communicate, the Activity LED on the back of the Xbox should blink rapidly. The Activity light will not blink at a regular rate but will do so in time with the Ethernet packets as they move across the network.

NEED TO KNOW... XBOX LIVE NETWORK CONNECTIONS



If you decide to network your Xbox to the Internet to use Microsoft's Xbox Live gaming service, you will probably be connecting through a *router*. A router is a device that routes your network traffic to the Internet. Like most other things, not all routers are created equal. Check out www.xbox.com/en-US/support/live/con-routers.htm to make sure that your router will work with Xbox Live.

One issue that can cause problems for users setting up networking and the using Xbox Live is the use of ports. The Transmission Control Protocol/Internet Protocol (TCP/IP) and the User Datagram Protocol (UDP) are two of the most common formats used for sending data packets between computers. This is the way that computers talk to each other over the Internet and on other networks. When a LAN is connected to the Internet, it is common to have a computer that acts as an intermediary. This intermediary computer is called a *firewall*. Some firewalls are also integrated into the router. The firewall passes allowed data packets between the Internet and the LAN while it stops packets that it doesn't know about. Part of what the firewall knows about a packet is the port number that the packet is assigned to. Port numbers are commonly assigned to specific applications; there are many standard port

numbers. For example, the HyperText Transmission Protocol (HTTP) is used for World Wide Web communications, no matter which browser a person uses to surf the Web. All HTTP transmissions use port 80. Therefore, a firewall would know that a packet with a port of 80 is going to be used by a Web browser program and will typically let that traffic pass into and out of the network.

When an Xbox is communicating with Xbox Live, it does so through the ports listed in Table 3.3. Any firewall between the Internet and the LAN that the Xbox is on must allow data packets using these ports. If your Xbox cannot communicate with Xbox Live and you are sure that it is set up correctly on the LAN, one of the first things to check is the port settings on the firewall. Make sure that these ports are “open,” which lets the firewall know that any packets bearing these port numbers are allowed to pass between the LAN and the Internet.

Table 3.3 Xbox Live’s Required TCP and UDP Ports

Protocol	Port
TCP	53
UDP	53
UDP	88
TCP	3074
UDP	3074

Creating Your Own Crossover Cable

If you’re just going to play some multiplayer games against one other player, sometimes setting up a whole network with multiple cables linking machines is overkill. Wouldn’t it be much simpler to just string one cable between two Xboxes and have the machines directly networked? The answer is “Yes!” But there’s a catch. A standard Ethernet cable will not work. The cable we need is called a *crossover cable*. It is so named because the normal connections are crossed to allow two machines to connect via the single Ethernet cable.

Crossover cables are found in many stores that carry networking supplies. However, most crossover cables are very expensive in comparison to standard “straight-through” cables. You can easily make your own crossover cable for less than half of what you’d typically pay at a retail store. Why buy something when you can make it?



Preparing for the Hack

For this hack, the only tool you will need is an RJ-45 Crimp Tool (Radio Shack part #279-405, \$33.99). You will also need the following materials:

- Two RJ-45 plugs (Radio Shack part #279-406, 5-pack \$4.19)
- An Ethernet cable (Radio Shack part #278-830, 100-foot bulk \$28.99)
- Colored electrical tape or permanent marker
- RJ-45 plug strain-relief boots, optional (Radio Shack part #278-1751, 4-pack \$1.29)

The Ethernet cable can be any length up to 320 feet (100m), although you'll probably want a length of 20 to 50 feet (6 to 15m), which is easier to deal with and is a convenient length for networking between two Xboxes in the same room or adjacent rooms.

The optional strain-relief boots are slide-on covers for the RJ-45 connector. As the name implies, they provide strain relief to the cable. By extending past the plug, the boots makes the cable slightly straighter where it enters the plug, thus lessening the strain placed on the cable if it is pulled or routed at some angle. Strain-relief boots also help protect the tiny, plastic retaining clips from breaking off the RJ-45 connector.

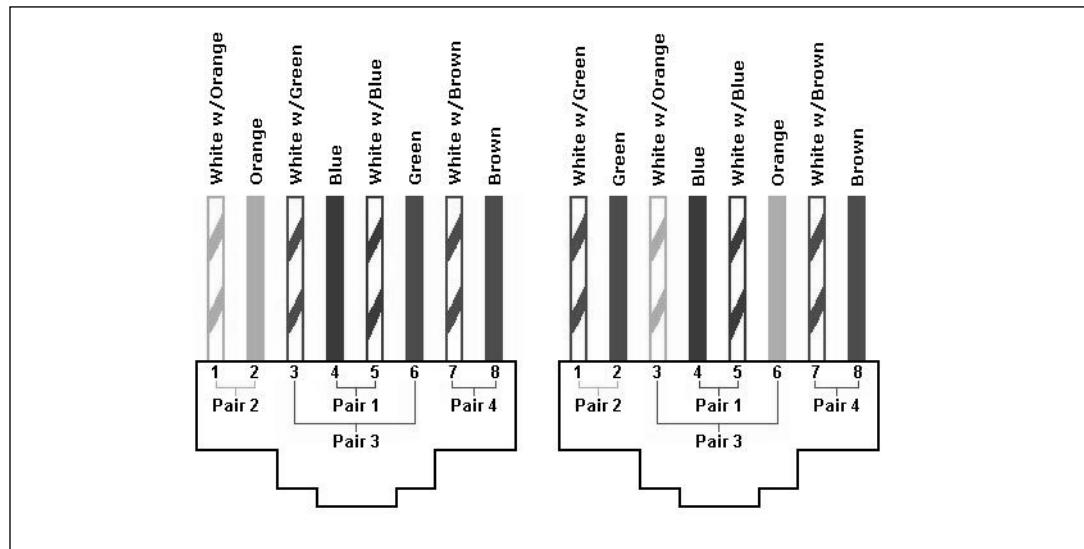
Performing the Hack

To create your crossover cable, perform the following:

1. Begin by stripping off a half inch (approximately 12mm) of the outer sheath. Slide the strain-relief boot onto the cable (if applicable). Separate the inner wires into the following order:
 - White with green trace
 - Green
 - White with orange trace
 - Blue
 - White with blue trace
 - Orange
 - White with brown trace
 - Brown
2. Insert the wires into one of the RJ-45 plugs, taking care to keep them in the proper order as seen from the side of the plug without the retaining clip.
3. Crimp the RJ-45 plug onto the wires using the crimp tool and push the boot into place (see Figure 3.29).

Figure 3.29 Crimping the RJ-45 Plug

4. Strip off a half inch (approximately 12mm) of the outer sheath from the other end of the cable. Slide the strain-relief boot onto the cable (if applicable). Separate the inner wires into the following order:
 - White with orange trace
 - Orange
 - White with green trace
 - Blue
 - White with blue trace
 - Green
 - White with brown trace
 - Brown
5. Insert the wires into one of the RJ-45 plugs, taking care to keep them in the proper order as seen from the side of the plug without the retaining clip.
6. Finally, crimp the second RJ-45 plug onto the wires using the crimp tool and push the boot into place. Figure 3.30 shows the correct wiring for both ends of your crossover cable.

Figure 3.30 Crossover Cable Wiring at each RJ-45 Plug

- Now that you've completed your crossover cable, mark it with a permanent marker or sticker so that it is easily recognizable and will not get confused with a standard, straight-through Ethernet cable. Colored electrical tape wrapped around the cable just behind each of the plugs is an inexpensive and easy way to mark the cable. Red electrical tape stands out well and is easy to find in hardware stores. Cable-labeling kits are also available at many electronics stores such as Radio Shack or Fry's Electronics.

Testing and Troubleshooting

To test the cable, plug it into the Ethernet sockets on the two Xboxes. The green Connection LED next to the Ethernet socket should illuminate, and you could see the yellow Activity LED flash intermittently if there is any network activity.

Then start up your favorite Xbox game that has a multiplayer option. Go to the Multiplayer menu and configure the options such as team colors or the game level to play on, as needed. When both Xboxes are ready to play the game, they should establish the network connection automatically. You should most definitely now see the Activity LED flashing as Ethernet traffic goes over the network.

If the Connection and Activity LEDs show no signs of life, the network connection has not been successfully established. Typically, this is due to an improper or incomplete crimp on the cable. Simply place each RJ-45 back into the crimp tool and apply pressure. Then try to establish the network connection again. If the cable still doesn't work, check the plugs to make sure the wires are in the proper order at each end, as shown in Figure 3.30.

If you start to make a lot of your own Ethernet cables, an inexpensive cable test set can save you a lot of frustration. A good Ethernet cable test set is the Ideal LinkMaster Tester model 62-200, available for about \$80.00. These test sets can be purchased at many electrical supply stores or home improvement centers such as Home Depot or Lowe's.

Extending the Network Status LEDs to the Front Panel

The Xbox contains a hardwired 10/100 Mbps Ethernet network port. Like most Ethernet ports on desktop computers, it is located on the back of the machine, where it is convenient to run the cables.

There are two integrated status LED lights with the Ethernet port; they indicate Link/Connection and Activity. The location of these lights is fine for when you first set up the box and want to know what the Ethernet status is when the Xbox is plugged in to the network. Unfortunately, if you have a problem later on, the location of these LEDs can become troublesome. To see both the lights and any on-screen messages, you have to pull the Xbox out of its location and turn it around so that you can see the back.

This hack extends the network status LEDs to the front panel, where you can see them more easily. Since the LEDs already exist on the Ethernet jack, it is a simple matter of extending the connections from the jack to the front panel and installing two miniature LEDs there.



Preparing for the Hack

The only components required for this hack are two miniature LEDs. They were obtained in a Radio Shack pack of 20 assorted LEDs (part #276-1622). Other LEDs will also work, but the miniature LEDs fit well on the Xbox's front panel. The tools required for this hack are:

- A Phillips screwdriver, size #1
- A soldering iron and solder
- A Dremel tool or drill
- Twenty or 22 gauge wire, four pieces, each approximately 12 inches long (preferably two red and two black)
- Electrical tape or a heat gun and heat-shrink tubing
- Hot glue or epoxy

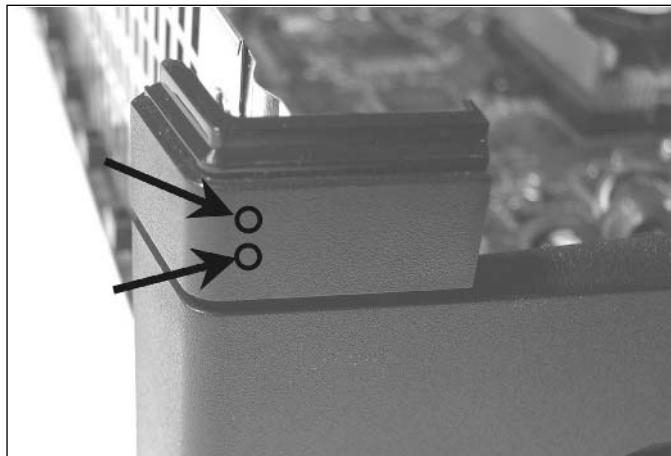
Performing the Hack

Perform the following:

1. Open the Xbox case and remove the front panel as described in the previous "Opening the Xbox" section.

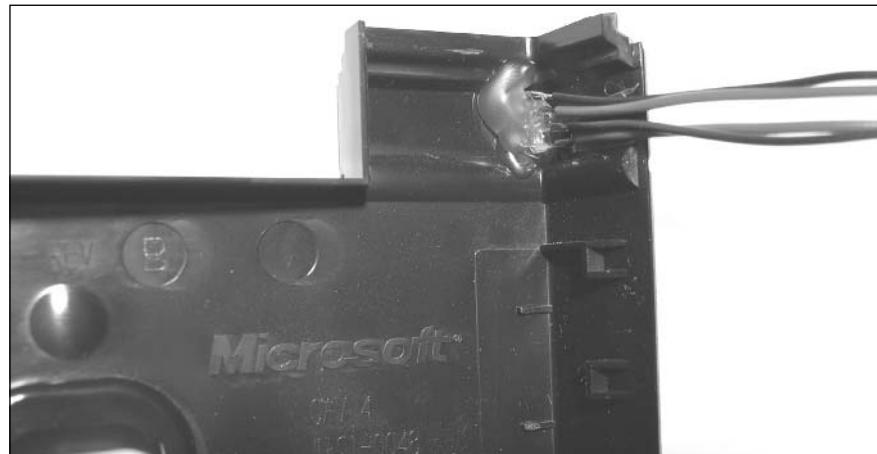
2. Drill two holes in the upper-left side of the front panel (see Figure 3.31). Each hole should be the right size to accommodate one LED. This location allows room for the LEDs and their wires and allows the LEDs to be seen from the front of the Xbox when it's operating.

Figure 3.31 Front-Panel Locations for the LEDs



3. Now solder two wires to the leads of the new LEDs. Be sure to observe the polarity of the LED. It is recommended to solder the black wire to the cathode of the LED (the side closest to the flat edge of the LED) and to solder the red wire to the anode of the LED (the remaining free pin on the LED). Insulate the connections using the electrical tape or heat-shrink tubing.
4. Using the hot glue or epoxy, secure each LED into place on the inside of the front panel. Make sure you can see the LED from the exterior (see Figure 3.32).

Figure 3.32 Back View of the LEDs Mounted on the Front Panel



- Put the front panel back into place and secure it by engaging the snap locks. Run the wires inside the case to the Ethernet port. Follow the path shown on the left side of Figure 3.33.

Figure 3.33 Run the Wires from the Front Panel to the Ethernet Port



- Solder the remaining end of the wires to the existing LEDs on either side of the port. Solder the leads from the yellow LED to the left side (as seen facing the Xbox from the front), and solder the leads from the green LED to the right side. Make sure that you observe the correct polarity of the LEDs.
- Finally, reassemble the Xbox by following the disassembly steps in reverse.

Congratulations! The hack is complete.

Testing and Troubleshooting

To test this hack, you will need a working network connection of some type, no matter if it's a standard network connection through a hub or just between another Xbox with a crossover cable. Plug one end of the network cable into the Ethernet port on the Xbox and the other end into the other device. With everything powered on, the green Link/Connection LEDs should light continuously, whereas the yellow Activity LEDs should flash intermittently as Ethernet traffic goes over the net-

work. The new network LEDs should light at the same time as the original LEDs on the back of the Xbox.

The most common problem when installing LEDs into a system is incorrect polarity. If one of the LEDs lights but not both of them, the polarity to the nonlighting LED is likely backward. Simply unsolder the two connections to the troublesome LEDs and reverse their locations.

Wireless Networking Hacks

Wireless networking, also known as 802.11b or WiFi, allows you to escape the physical constraints of wired networking. When used in gaming, WiFi can simplify the process of adding new Xboxes to an existing network, since you don't have to run additional wires to connect everything together.

Adding a Wireless Networking Adapter to the Xbox

Adding a wireless networking adapter to the Xbox is usually a matter of purchasing whichever adapter best fits your needs, plugging it into the Xbox, configuring it, and using it. You will of course need a pre-existing wireless network.

NEED TO KNOW... WIRELESS NETWORKS



Setting up a wireless network is beyond the scope of this book, so if you don't have one but would like to set up your own, you'll have to do some additional research. There are several great resources both in print and on the Web:

- *Designing a Wireless Network*, Syngress Publishing, ISBN: 1-928994-45-8
- *Jeff Duntemann's WiFi Guide, Second Edition*, Paraglyph Publishing, ISBN: 1-932111-88-3
- *Hack Proofing Your Wireless Network*, Syngress Publishing, ISBN: 1-928994-59-8
- *Ultimate Wireless Reference CD*, Syngress Publishing
- *Installing Troubleshooting and Repairing Wireless Networks*, McGraw-Hill Publishing, ISBN 0-07-141070-8
- *802.11 Wireless Networks: The Definitive Guide*, O'Reilly Publishing, ISBN 0-596-00183-5
- NetStumbler Forums, <http://forums.netstumbler.com>
- Tom's Hardware Forums, www.tomshardware.com

Figure 3.34 shows several popular wireless networking devices: two by Linksys and one by Microsoft.

The Linksys WET11 is technically a “wireless bridge” and is designed to adapt any Ethernet device to a wireless network. It is not designed specifically to make a game console wireless, although it will work for this purpose. The Linksys WGA11 is designed to adapt game consoles to wireless networks. It will work with any Ethernet-enabled videogame console, not just the Xbox. Both Linksys units come in 11 Mbps and 54 Mbps models. These are compatible with the two standard speeds of 802.11b. Both adapters also require a separate desktop computer on the network in order to configure the device. (Unfortunately, configuring the devices cannot be done straight from the game console.)

Microsoft’s Xbox Wireless Adapter was designed to work with the Xbox but will not work with other consoles. On the plus side, since it is made specifically for the Xbox, it only needs an Xbox to get it running, since configuration is done completely through the Xbox dashboard. No separate computer is needed to configure it.

Figure 3.34 The Linksys WET11, Linksys WGA11B, and Microsoft Xbox Wireless Adapter



To add a wireless adapter to the Xbox, configuration is an important process. Issues such as whether games will be played over the Internet or merely on a standalone network influence any design since different equipment will be required (such as routers, hubs, or DSL or cable modems). In addition to such considerations, wireless networks pose their own challenges over a standard wired network. For example, most wireless networks have the ability to use encryption. This prevents a casual passerby from picking up data that is being broadcast on your wireless network. If this is a gaming-only network, you may not care that someone outside your home might detect the broadcasts. However, if you are adding the Xbox to a wireless network that is part of a business that may have confidential data traveling on the network, then encryption could be a very important requirement.

When setting up the wireless network, you have to know some basic terms. If your wireless network has an access point (AP) or router as a central connection, then the network is known as an *Infrastructure* network. If there is no central AP or router, then the wireless connections talk to each other in what is known as an *Ad Hoc* network. While these terms are those generally accepted for

wireless networking, they aren't always used. For example, in the setup instructions for the WGA11, Linksys refers to Ad Hoc networks as *Head to Head Gaming*.

Adding a Removable Antenna to the Microsoft Xbox Wireless Adapter

The Microsoft Broadband Networking Xbox Wireless Adapter (Model MN-740; see Figure 3.35) is a decent solution for a low-range wireless network if your console is close to the wireless access point (WAP). However, if you're any distance away from the AP, as you might like to be to take advantage of being on a wireless network, the MN-740 will need a better antenna. This hack shows you how to remove the stock antenna from the Xbox Wireless Adapter and replace it with a higher-gain antenna.

Figure 3.35 The Xbox MN-740 Wireless Adapter with Original Antenna



NEED TO KNOW



The original idea for this modification came from Chris Arocha of Florida. He figured out how to modify a Microsoft USB Wireless Network Adapter model MN-510 to accept a higher-gain antenna. The MN-510 is internally similar to the MN-740, and it was relatively easy to adapt Chris's design to the Xbox Wireless Adapter, which is specifically for the console. You can see Chris's original hack, "Hacking the Microsoft MN-510 USB Adapter," at <http://forums.netstumbler.com/showthread.php?t=11051>.



Preparing for the Hack

The components required for this hack are:

- Times-Microwave LMR-100A coaxial cable or equivalent. Approximately 6-inch length (about \$0.35 per foot)
- Reverse polarity SMA bulkhead female socket (RP-SMA) for LMR-100A cable (approximately \$6.00)

Both of these components can be purchased from Fleeman Anderson & Bird Corp. (www.fab-corp.com.)

The tools required for this hack are:

- A crimp tool for LMR-100A cable. This tool can also be purchased for around \$55.
- A soldering iron and solder
- A Phillips screwdriver, size #1
- An X-ACTO knife or hobby blade
- A Dremel tool (optional)

Performing the Hack

Perform the following:

1. Open the Xbox Wireless Networking Adapter by unscrewing the two Phillips screws on the bottom of the unit (see Figure 3.36). They are located underneath the label.

Figure 3.36 The Bottom of the Xbox Wireless Adapter Showing the Case Screws



- Once the screws are removed, slide the top cover to the front. The top cover will then lift off. The opened Xbox Wireless Networking Adapter is shown in Figure 3.37.

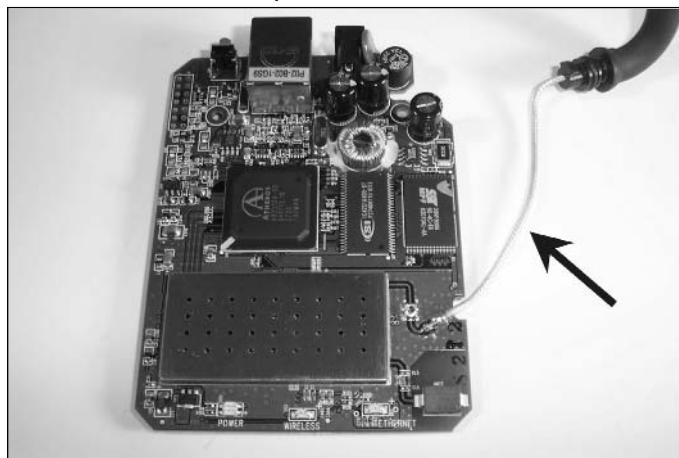
Figure 3.37 Xbox Wireless Adapter with Top Removed



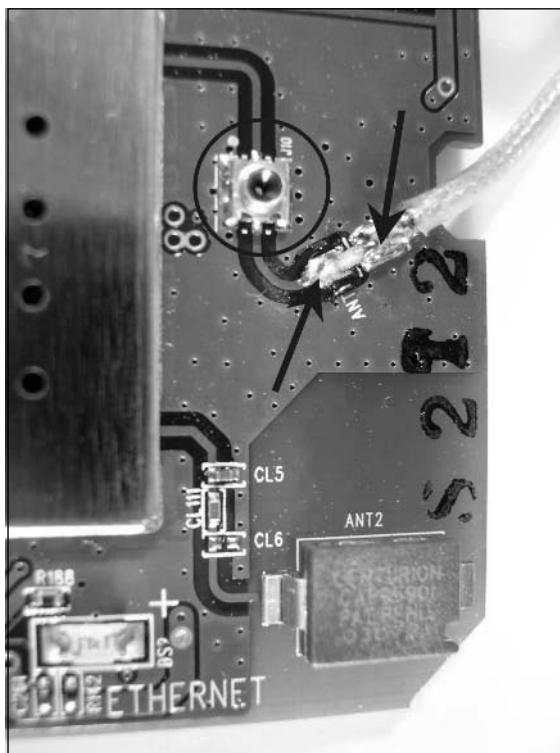
- Two small clips at the rear hold the circuit board in place (see Figure 3.38). Pushing the clips slightly will move them out of the way, and the circuit board will be released from the base (see Figure 3.39). The stock antenna is connected to the PCB via a short antenna lead. This is what we'll replace.

Figure 3.38 Clips to Hold the Circuit Board



Figure 3.39 The Xbox Wireless Adapter PCB

4. Now remove the stock antenna cable by unsoldering it from the two points on the circuit board (see Figure 3.40). The first point is where the center conductor of the antenna cable is soldered to the board, and the second is where the outer shielding is soldered to the board.

Figure 3.40 Unsolder the Antenna Cable

5. Next, create a new cable assembly using the 6-inch LMR-100A cable. Follow the instructions on assembling the RP-SMA connector to the cable. If your connector didn't come with instructions, you can find instructions for the RP-SMA connector and many other popular connectors and cables at www.TimesMicrowave.com. The completed cable can be seen in Figure 3.41.

Figure 3.41 The Completed Cable



6. Once the RP-SMA connector is attached to the cable, strip back the opposite end of the cable. Apply a small amount of solder to the inner conductor and the outer shield. This process is known as “tinning.”
7. Solder the center conductor of your new cable to the point on the circuit board marked ANT on the silkscreen.
8. Solder the outer shield to the circuit board where the original shield had been soldered, just behind the ANT connector. Make sure that the outer shield connection is not shorting to the center conductor. If it is, the hack will not work.
9. You will need to remove some plastic from the case to mount the new connector jack. Do this on the inside area, around the opening where the old antenna came through. You can grind this away using the Dremel tool or by cutting it with a hobby knife. Perform this process for both the top and bottom halves of the case. You should test-fit the connector and cable assembly to the case as you go. This will prevent you from removing too much material.
10. Reassemble the Xbox Wireless Networking Adapter by placing the two halves of the case together and screwing the two screws back into place.
11. Finally, you can add an external 2.4 GHz antenna (a slightly more powerful antenna than the original is shown in Figure 3.42). Enjoy your extended range of your wireless adapter!

Figure 3.42 The Xbox MN-740 Wireless Adapter with a New, More Powerful Antenna

If you are ambitious enough, you might want to consider making a house-to-house long-range wireless link with your gaming buddies. Adding an external antenna similar to the one shown in Figure 3.43 makes this a possibility. You are limited only by your imagination and any radio regulations in your locale.

In the United States, the Federal Communications Commission (FCC) has regulatory authority over radio transmissions, including those used in wireless computer networking. Specifically, Part 15 of the FCC regulations covers unlicensed radios used in wireless networking. If you do decide to add an antenna to your Xbox Wireless Adapter, make sure that your setup does not exceed the regulation detailed in Part 15. For more information, go to the FCC's Web site at www.fcc.gov.

Figure 3.43 An Exterior Antenna Makes Long-Range Gaming a Possibility

Under the Hood: How the Hack Works

This hack replaces the stock antenna with a popular antenna connector, the RP-SMA. The RP-SMA works well with the radio frequencies involved in WiFi (2.4GHz), it fits several different brands of antennae, and its size is easily accommodated into the case of the Xbox Wireless Networking Adapter. Putting in this connector allows you to plug any WiFi (802.11b or 802.11g) antenna equipped with the RP-SMA mate into the Xbox Wireless Networking Adapter. Alternately, different antennae may be used via an adapter cable. Feel free to experiment with various antenna types, which could result in higher (or lower) gain, better (or worse) wireless range, and more (or fewer) directional characteristics.

NEED TO KNOW... ANTENNA JACK



Those readers who are familiar with radio equipment no doubt have noticed what appears to be a tiny antenna jack on the circuit board of the Microsoft Broadband Networking Xbox Wireless Adapter (circled in Figure 3.40). The logical question is, "Why not use that jack?"

The reason we don't use it is that the plug to fit the jack in question is both hard to find and very fragile. The socket and plug are Hirose MS-156 connectors. The Hirose MS-156 series is designed to be a test connector and, as such, is rated with a life of 100 maximum insertion and removal cycles, which is very low. These connectors are so fragile that some people have been known to break them on the first insertion of an antenna. These connectors also have limited distribution to the general public, making a replacement difficult to obtain if you do damage one. For our hacking purposes, it just makes more sense to unsolder the existing cable and connect our new custom cable.

Installing a Modchip

The goal of this hack is to install a *modchip* into your Xbox. Because a modchip is able to operate and interface with the Xbox's hardware at some of the lowest hardware levels, it will allow you to gain the ultimate control of your Xbox. With the modchip installed in your Xbox, you will add different capabilities, such as alternate graphic interfaces or even alternate operating systems such as Linux.

A Brief Introduction to Modchips

When a viable set of exploits and workarounds is found to bypass a particular security feature, the modchip is born. Modchips are small printed circuit boards (PCBs) with wires that attach to various components on the console's main board. A modchip is usually controlled by a Microchip PIC or standard programmable logic such as a programmable logic device (PLD), but modern modchips for the Xbox and other systems include an FPGA and Flash ROM so that they can be updated with bug fixes and new features.

Many different types of modchips exist for the Xbox, all with specific functions they are designed to perform. Before laying down your hard earned cash to purchase a modchip, make sure you understand the capabilities of the device. Additionally, some modchips are very dependent on the version of

the Xbox hardware they will operate on, so be sure that your desired modchip will even work with your hardware. (See the “Xbox Versions” section at the beginning of this chapter for more details.)

Typically, a basic Xbox modchip will load an alternate startup routine into the Basic Input/Output System (BIOS) of the Xbox. The BIOS controls the low-level input and output functions of the Xbox. Using a customized BIOS will allow us to load functions that are not normally part of the standard Xbox user interface.

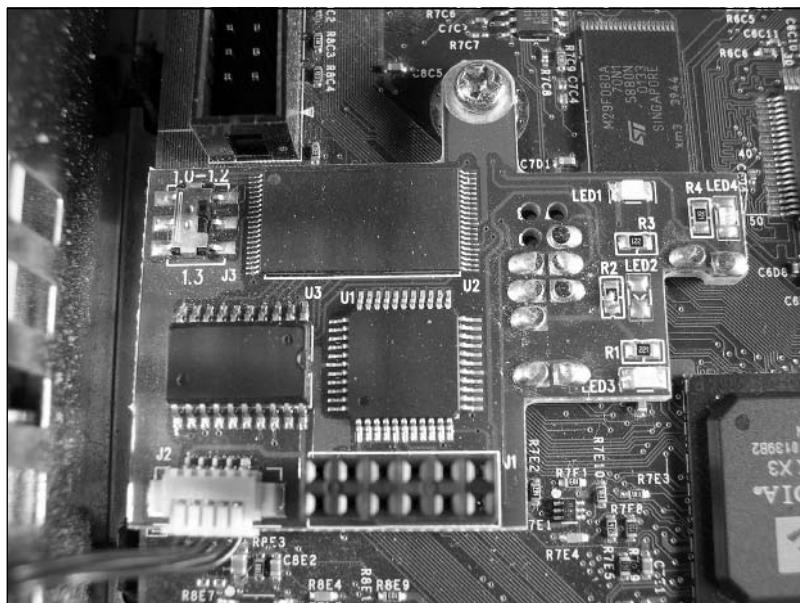
NEED TO KNOW... MODCHIP ISSUES



Circumventing copy protection measures (for example, by using modchips) in order to play pirated or unauthorized games may be unlawful in your country. Both U.S. and other governments have successfully prosecuted sellers of modchips and similar devices. Please check with your country's laws regarding such activities. In addition, downloading and using BIOS images that copy Microsoft's proprietary code infringes Microsoft's copyrights.

Microsoft's gaming service, Xbox Live (or XBL, as it's often called), checks for the presence of modchips on the Xbox every time you check in with Xbox Live. If you attempt to use Xbox Live with a modified Xbox, the service will note the serial number of your Xbox and permanently ban that Xbox from XBL. Removing or disabling the modchip will not reverse the ban. The only way to get back on Xbox Live is to use a different Xbox. The main reason behind the XBL policy is that it prevents cheating. Due to the fact that modchips allow software modification to the games being played, allowing a modified Xbox onto the service would put those players with unmodified Xboxes at a disadvantage.

The modchip used in this section is the Xecuter Lite+ from www.teamxecuter.com (see Figure 3.44). At the time of this writing, version 2.3B is now available and retails for approximately \$50.00. It can be purchased from many online sites, including www.system-mods.com. As of September 2004, production of the Xecuter version 3.0 is coming shortly.

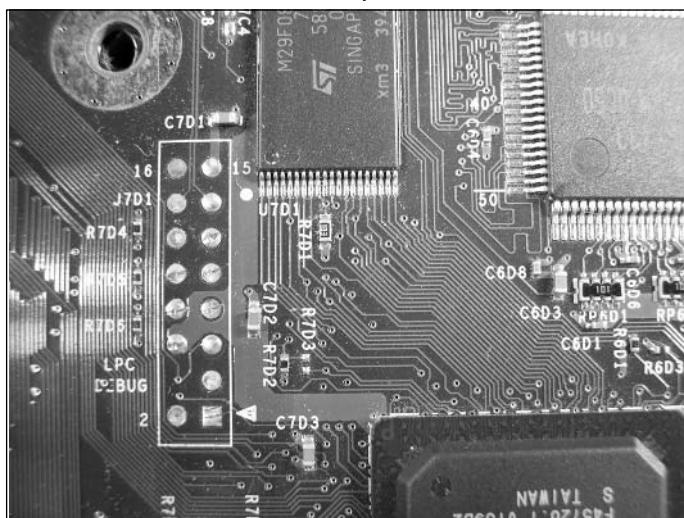
Figure 3.44 The Xecuter Lite+ Modchip

Modchips are installed into the Xbox in a variety of ways. Some must be soldered directly to the Xbox motherboard. Others are “solderless” in that they use “pogo pins” to make contact with test points on the Xbox motherboard without the need to solder. Pogo pins are commonly used for product testing during manufacturing, since they can quickly make contact with the circuit under test. The pogo pin contacts are spring-loaded and look and act like miniature pogo sticks. The spring-loaded design keeps the pin in contact with a contact pad on the motherboard, thus ensuring a good electrical connection. The pogo pins on the underside of the Xecuter Lite+ modchip are shown in Figure 3.45.

Figure 3.45 Pogo Pins on the Bottom of the Xecuter Lite+ PCB

No matter what the specifics of the modchips, they almost all function via the Low Pin Count (LPC) data bus. The LPC bus is an industry-standard interface designed by Intel (and commonly used on PC-based architectures) that replaces the aging Industry Standard Architecture (ISA) interface. The LPC bus can be used to connect the CPU to other peripheral devices such as the BIOS, mouse, and keyboard. On the Xbox, the LPC bus is used for testing and debugging during production. It consists of 15 contacts in an 8x2 array. The LPC bus is located on the Xbox on the left side of the motherboard as you face the front of the Xbox (see Figure 3.46). You should note that Pin 4 is used only as a key, and the contact is completely missing from the Xbox's motherboard.

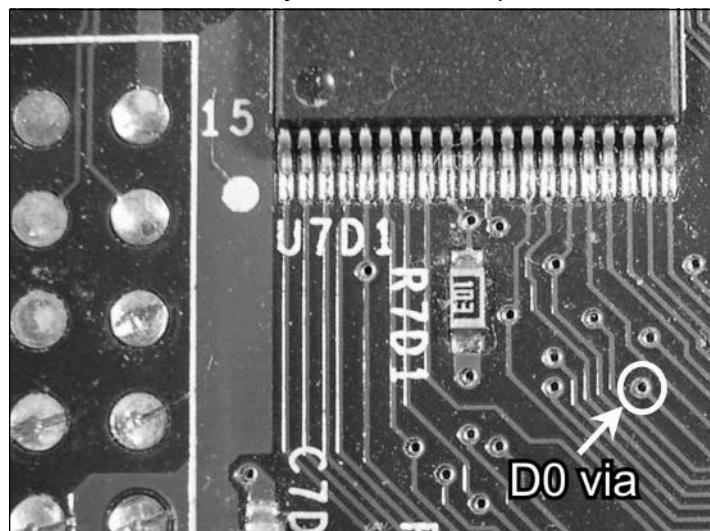
Figure 3.46 The LPC Bus: A Common Modchip Interface



Modchips function by loading an alternate version of firmware or “boot code” into the Xbox at system startup. By design, the Xbox looks to load alternate firmware from the LPC if the normal boot code is not available from the Flash ROM on the Xbox. The Xbox is tricked into believing that the normal Flash ROM is not available if the lowest bit on Flash ROM’s data bus line, D0, is pulled to ground (0V). In other words, when D0 is forced to a low state, the Xbox loads alternate boot code from the modchip instead of from its standard Flash ROM.

The D0 line is accessed by means of a *via* to the right of the LPC contacts (see Figure 3.47). A via is a plated through-hole connecting different layers of a PCB. Some brands of modchips require that a small wire be soldered to the D0 via (the protective green soldermask coating will first need to be scratched off the via using an X-ACTO knife). Other modchips use pogo-pin contacts. Due to the small size of the via, those modchips that use pogo pins need to be very precisely aligned to function.

Figure 3..47 The D0 Data Line Necessary for the Modchip to Function



Preparing for the Hack

For this hack, you will need to obtain a modchip. We're using the Xecuter Lite+ for this chapter. You will also need the following tools:

- A Torx T-20 screwdriver (to open the Xbox)
- A Torx T-10 screwdriver (to open the Xbox)
- An X-ACTO knife or hobby blade (to open the Xbox)
- A Phillips screwdriver, size #2
- A soldering iron and solder (optional, depending on the type of modchip you are using; for the Xecuter Lite+, shown in Figures 3.44 and 3.45, you don't need a soldering iron since it connects directly to the Xbox circuit board with pogo pins)

Performing the Hack

Perform the following:

1. Open the Xbox case as described in the “Opening the Xbox” section. You do not need to remove the front panel.
2. Plug one end of the cable assembly (received with your modchip) into the Xecuter main circuit board.

3. Locate the LPC bus on the left side of the motherboard. Adjacent to Pin 16 of the LPC, there is a Phillips-head screw holding the motherboard in place. Remove the screw and set it aside.
4. Place the Xecuter modchip on the motherboard, noting exact alignment of the pogo pin for the D0 via (see Figure 3.44). Note that the D0 line has been moved slightly in versions 1.2 and higher of the Xbox motherboard. Make sure that you know the exact location of the via for your version of the Xbox or the modchip will not function properly. Complete installation tutorials for the Xecuter modchips can be found at www.teamxecuter.com.
5. Replace the motherboard screw, passing it through the Xecuter's circuit board to secure it into place. Your Xbox should resemble the one shown in Figure 3.48.

Figure 3.48 Xbox with Modchip installed



6. Next, mount the Xecuter's switchboard on the front of the Xbox. The ideal placement is on the lower-left front corner, below the DVD drive (see Figures 3.49 and 3.50).

Figure 3.49 Xecuter Switchboard Installed on the Bottom of the Xbox

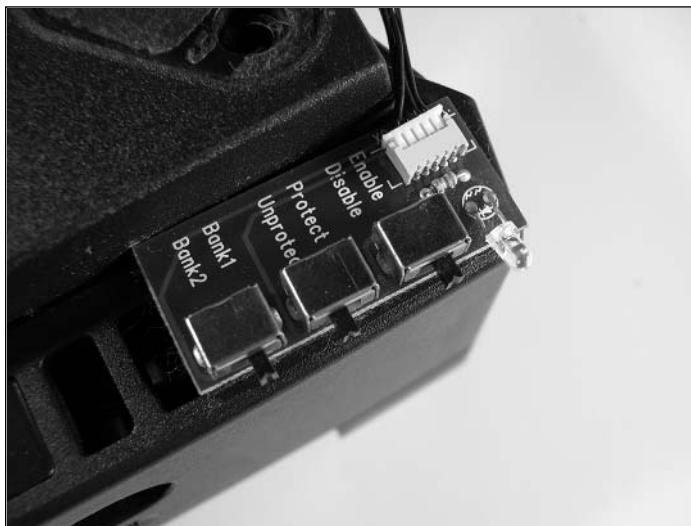


Figure 3.50 Xecuter Switchboard as Seen From the Front of the Xbox



7. Now route the cable assembly (connected to the modchip) to the outside of the case. The cable will need to pass through the interior metal RF shield and the Xbox's plastic housing.
8. Reassemble the Xbox by following the disassembly steps in reverse.
9. Plug the other end of the cable assembly (received with your modchip) into the Xecuter switchboard on the outside of the Xbox.

The physical installation of your Xbox is now complete. For the modchip to operate, you will need to obtain a boot disk image and desired firmware. The firmware is usually

included with the modchip, but if it isn't, or if you want updated or special firmware, follow the next steps.

10. Using an Internet Relay Chat (IRC) client, connect to an EFNet server (<irc.efnet.org>) and join either the #xbins4newbies or #xbins channel. (For more information on IRC, visit www irc org.) Popular IRC clients include mIRC for Windows (www mirc com), BitchX for UNIX (www bitchx org), and Conversation for Mac OS X (<http://homepage mac com philrobin conversation>).
11. Upon joining the channel, you should receive a message similar to this:

```
-onjoin- Welcome to #xbins! In Order to get FILES you must type /msg xbins !list he will then give you FTP info along with a username/password. You MUST be in this channel in order to recieve any info/files. If he is not in the channel then a server split is occurring and he will be back shortly. #xbins is not a help channel, it is a distribution channel. If you would like to talk in general, join #xbins-chat. If you need help, join #xbins-help.
```

```
-onjoin- For the latest releases type /msg onjoin !releases (for the 3 most recent) or !releases 10 (for the 10 most recent). Need to find a file on the FTP? Use SITE SEARCH as a raw command once logged into the ftp (CTRL + R for flashfxp)
```

12. In the channel, send the */msg xbins !list* command. You should receive a private message from the *xbins* user similar to the following, which will contain the distribution FTP site, username, and password (note that the actual username and password have been deleted from this example):

```
<xbins> FTP ADDRESS: distribution.xbins.org PORT: 21 USERNAME: username  
PASSWORD: password NOTE: This Username And Password will be deleted upon connection for security reasons. This site contains 100% homebrew files and absolutly NO warez. Brought to you by #xbins and team xecuter  
<xbins> Each person is allowed 30 files a day. We do NOT tolerate GREED and you shall be banned if you break this rule. Got a ACCESS DENIED error? DON'T use IE or LeechFTP. Use FlashFXP/SmartFTP/CuteFTP for best results. AFTER EVERY ATTEMPT, SUCCESSFUL OR NOT, YOU NEED TO !list AGAIN.  
<xbins> If you would like CVS builds of XBMP/XBMC or any other XBMP/XBMC releases, get it on our ftp at xbmp.uk.xbins.org (Europe 100mbit) Username = username Password = password
```

13. Armed with this information, use an FTP client to connect to the given address and log in with the given username and password. You can only log in one time with each username/password combination before it is deleted. Upon connection, you will be prompted with a list of directories: PC, DEBUG, and XBOX.
14. Download the desired BIOS and boot disk files from the XBOX directory. The files you download are specific to your modchip and the goals of your hack. Make sure that the

BIOS image fits the size of the image bank(s) in your model of Xecuter (or other modchip). For this hack, we're using the Evolution X, or Evo X, boot disk.

With the firmware and boot disk images in hand (either obtained with the modchip or from the previous steps), we can now program the modchip and configure the Xbox. Follow these steps:

15. Using your CD burner of choice, write the BIOS image to a CDR.
16. Using your CD burner of choice, write the boot disk image to a CDR.
17. Boot the Xbox using the new boot disk.
18. When prompted for the BIOS image, swap the boot disk for the one containing the BIOS image. You will then be prompted to transfer the BIOS image to the modchip.
19. Next, transfer the required boot and operating system files to the Xbox's hard drive. These will depend on the boot disk image you selected.

The SlaYer Xbox Auto-Installer from <http://slayer.xbox-scene.com> is a great tool for installing the needed files for a modchip. In addition to the actual auto-installer program, the SlaYer site has an excellent instruction manual detailing all the steps of the installation process and several variations. These variations include upgrading hard drives in the Xbox and upgrading the BIOS.

20. Once the BIOS has been loaded into the modchip and the system files have been transferred to the Xbox hard drive, the Xbox can be shut down.
21. Ensure that the leftmost switch of the Xecuter's switchboard is set to Enable, and restart the Xbox. If the Xecuter is installed correctly, the LED on the switchboard should illuminate and the Evo X splash screen will be displayed on the TV (see Figure 3.51). The Evo X is very similar to the standard Xbox boot screen, except that it has the Evo X logo in the upper-left corner.

Figure 3.51 Evolution X Boot Screen



22. Once the boot-up sequence is completed, a new dashboard should load. The dashboard is Xbox's graphical user interface. Figure 3.52 shows the standard Xbox dashboard; Figure 3.53 shows the Administr8or dashboard, just one of several alternative dashboards that come as part of the Evolution X package.

Figure 3.52 Standard Xbox Dashboard



Figure 3.53 Evolution X Administr8or Dashboard



Congratulations! If you can see the new dashboard, your modchip is successfully installed and operating correctly.

Running Linux on an Unmodified Xbox

Linux, the popular open-source, UNIX-based operating system (www.linux.org), can be installed directly onto an unmodified Xbox due to a bug in the Save Game feature of two Xbox games, 007: Agent Under Fire and MechAssault. You only need to use the game once to install Linux.

Unlike most of the other hacks in this chapter, this is more of a software hack than a hardware hack. No cutting, drilling, or soldering is required.

NEED TO KNOW... XBOX LIVE



The Save Game bug that enables this hack to work on an unmodified Xbox will not work if your Xbox has ever connected to Xbox Live. Once in contact with Xbox Live a new BIOS patch is uploaded to your Xbox. One of the fixes contained in this patch is a change that prevents the Save Game bug from working. The connection to Xbox Live may have occurred anytime your Xbox was on a network, even if you are not a subscriber to Xbox Live.



Preparing for the Hack

For this hack, you'll need the following materials:

- An unmodified Xbox (no modchip installed)
- A PC or Xbox already running Linux; it must have source code and development tools installed
- Ethernet-based network
- Original Xbox Game—007: Agent Under Fire or MechAssault (note that the rereleased Platinum Editions of these games have been patched to fix the bug used in this hack; also, 007: Nightfire will not work although it can easily be confused with 007: Agent Under Fire)
- USB keyboard
- USB thumb drive
- USB mouse (optional)
- USB four-port hub (optional)
- USB-to-Xbox adapter cables, one for each USB device; these are available on eBay or from Lik-Sang (www.lik-sang.com/info.php?category=83&products_id=2154) and many other online videogame retail stores

Both the Xbox and your Linux-based PC must support the USB thumb drive. A great resource for checking which thumb drives work and which don't can be found at <http://unmodded.mine.nu/docs/XboxUsbCompatibilityList>. Here is a list of some of the more commonly used devices:

- Belkin 32MB USB Memory Stick
- Creative Labs Nomad Muvo MP3 Player 128MB
- EasyDisk 32MB
- EasyDisk 128MB
- Fujitsu Siemens 64MB Menustick
- Fujitsu Siemens SB-512 MemoryBird USB-2.0
- Gateway 16MB USB Flash Drive
- IBM 32MB USB Memory Key
- Kingston DataTraveler 32MB
- Lexar JumpDrive Secure 128MB
- Lexar JumpDrive Secure 256MB
- Lexar JumpDrive Sport 64MB
- Linksys Instant USB Disk 64MB
- Memorex 64MB
- Memorex ThumbDrive 256MB
- Minolta DiMAGE F300 with 64MB SD-Card
- PNY 64MB
- SanDisk Cruzer Mini USB 128MB
- Sandisk Mini Cruzer 256MB USB 2.0
- Samsung 64MB USB disk
- Sony 64MB MicroVault USB 2.0

Performing the Hack

Perform the following:

1. If you do not have the Xbox File Allocation Table system (FATX) currently on your Linux PC, you will need to install it. Installing this system on the Linux PC allows it to read and write files in the format expected by the Xbox. FATX can be installed as a Concurrent Versions System (CVS) file from the cvs.sourceforge.net server for Linux Kernel 2.4.21 or as a

patch file for the 2.4.20 version kernel. You can obtain FATX patch files from http://prdownloads.sourceforge.net/xbox-linux/kernel-2_4_20-0_7_0.patch.gz?download. Once your Linux PC is capable of recognizing the FATX file system, you're ready to continue the hack.

2. Now download the appropriate installation packages for Linux. These will vary depending on whether you have 007:Agent Under Fire or MechAssault. For 007:Agent Under Fire, you will want the 007linux.zip file currently available from www.xbox-linux.org/down/007linux.zip. For MechAssault, you will want the MechInstaller package currently available from www.sourceforge.net/projects/xbox-linux. Make sure you read the included README files to get the correct installation information.
3. You have a choice of installing the files over your network, if you have one, or by using a CD. Save the files to a location accessible on your network or burn the files onto a CDR. If you want to install the files via your Ethernet network, then choose the desired network share when prompted during the installation for the files' location.
4. Now, on your PC, unpack the files containing the “game saves” and copy them to the USB thumb drive. There should be several of these for booting the Xbox in various ways. At this time, we’re only interested in the installation file.
5. Remove the thumb drive from the PC and insert it into the Xbox using one of the USB-to-Xbox adapter cables.
6. Turn on the Xbox and go to the standard Xbox dashboard (see Figure 3.52). Choose the **Memory** menu.
7. Copy the saved game files from the thumb drive to the Xbox.
8. Load 007:Agent Under Fire or MechAssault as you normally would and wait for the game to start.
9. From the game’s Main menu, select the **Load Game** option and point to the Xbox hard drive as the location of the saved game.
10. Choose the saved game that you just copied to the Xbox. At this point the Xbox will crash. After several moments, a bootloader program will run, and you should see some information displayed on the screen, including the Xbox’s hard drive size. This will occur automatically.
11. Plug in your USB keyboard and mouse (if applicable). If you have decided not to use a mouse, the regular Xbox controller can serve to move the cursor around on the screen.
12. Linux will now start with its normal boot process. The BusyBox miniature kernel will start, prompting you for a login name and password. Login as **root** with a password of **xbox**.
13. Now that the miniature kernel/bootloader is running, you can choose to install Linux to the Xbox via your Ethernet network or by loading the installation CD. Follow the appropriate instructions in the README files for whichever method you have chosen.

That's it! Enjoy your new Linux-based Xbox!

Other Hacks

The hacks presented in this chapter are only the beginning of what you can do with your Xbox. We've listed some additional hacks here, and dozens (if not hundreds) more can be found on the Internet:

- **Hacking the Xbox, by Andrew “bunnie” Huang, ISBN 1-59327-029-1 (No Starch Press, 2003)** This is *the* definitive book for anyone interested in the art and science of reverse engineering. Bunnie takes you through a step-by-step process of analyzing the Xbox's security mechanisms. An absolute must-read for anyone interested in game console hacking!
- **Loading Linux onto the Xbox** A popular hack with most any game console, porting a UNIX-variant (such as Linux) to the Xbox has been an ongoing challenge for many developers. Besides our “Running Linux on an Unmodified Xbox” section in this chapter, you'll find Web sites and articles available on the subject, including:
 - **“Installing Debian on Your Unmodified Xbox,” 2600 Magazine, Spring 2004** This is a variation of installing Linux onto an unmodified Xbox. Some of the files are no longer available in the locations cited, so you might have to do some searching on the Internet to find everything you need.
 - **Xbox-Linux Project Web sites**, www.xbox-linux.org and www.sourceforge.net/projects/xbox-linux.
- **Creating a USB-to-Xbox Cable, www.xbox-linux.org/Xbox_Linux_USB_HOWTO** Adding a standard USB device to your Xbox isn't difficult, and there are many ways to create a USB-to-Xbox interface. This page provides useful information on how to do so as well as resources telling where to buy pre-made cables if you are so inclined.
- **Adding Standard USB Ports to Your Xbox, www.xbox-scene.com/articles/two-usb.php** This hack guides you through adding two USB ports to your Xbox, making it easier to plug standard USB devices into the Xbox without using a USB-to-Xbox cable.
- **Adding a Second Hard Drive to Your Xbox, www.llamma.com/xbox/Mods/extraharddrive.htm** You can never go wrong with additional hard drive space. This hack guides you through installing an external hard drive to the Xbox.

Homebrew Game Development

Since the Xbox is still being produced and marketed by Microsoft, many commercial developers are still producing games for the console. However, there is still a small community of homebrew game developers who want to write their own games for the Xbox.

Currently, you have few ways to develop your own Xbox games. You can apply to Microsoft to become a licensed developer. However, unless you are associated with a company that is actively developing games for the retail market, this is not a realistic option. Becoming a licensed developer requires that you sign a nondisclosure agreement with Microsoft and purchase and use the company's proprietary Xbox Development Kit (XDK). This certainly is not a good solution for hobbyists.

If you search the Internet for Xbox development tools, you will no doubt see information on obtaining the XDK from sources other than Microsoft. Many copies of the XDK available online are pirated and, as such, should not be used. Be very cautious about purchasing or downloading an XDK from unauthorized sources. The good news is that there are several legal alternatives to Microsoft's XDK if you want to create your own games and applications for the Xbox.

The following links are good starting points for Xbox homebrew game development:

- **Official Xbox Developer Information, www.xbox.com/en-us/dev** Microsoft's official site for potential Xbox game developers. This site will also be of interest to anyone interested in the development and licensing of third-party peripheral products for the Xbox. The Xbox Incubator Program is designed to help smaller development teams obtain the necessary Xbox development tools (www.xbox.com/en-us/dev/incubator.htm).
- **X-Factor Development, www.xfactordev.net/index.php** Geared toward people who have the official XDK, this site provides news, code samples, an active discussion forum, tutorials, downloads, and links.
- **Cxbx, The Xbox Emulator, www.caustik.com/cxbx** Cxbx is an Xbox emulator for use on Windows-based PCs. It is an ambitious and ongoing project.
- **OpenXDK, <http://sourceforge.net/projects/openxdk>** Described as "an open-source, free, legal Xbox development kit," OpenXDK was designed specifically to enable hobbyist and homebrew development for the Xbox. To use OpenXDK, you must have Microsoft's Visual Studio.
- **fr*shgloop, www.trynamic.com** This site provides news, articles, and information on various homebrew game development projects. Source code is also available for selected homebrew games, which serve as indispensable reference material if you are considering writing your own games for the Xbox.

Xbox Resources on the Web

There are a great number of resources on the Web for enthusiasts of the Xbox. Whether you're looking for information about your favorite games or how to hack your system to give it capabilities never intended by the designers, you're sure to find it online somewhere. The following is a list of some of our favorite Xbox sites:

- **Microsoft's Official Xbox Site**, www.xbox.com
- **Official Xbox Magazine**, www.officialxboxmagazine.com
- **XBOX365**, www.xbox365.com
- **TeamXbox: Insider's Choice for Xbox Information**, www.teamxbox.com
- **XboxAddict**, www.xboxaddict.com
- **Xbox-Scene**, www.xbox-scene.com
- **Xbox-HQ**, <http://xbox-hq.com>
- **Xbox-Hacker**, www.xbox-hacker.com
- **Xbox-Saves Manager**, www.xbox-saves.com
- **#xboxhacker IRC (Internet Relay Chat) Channel** Located on the EFnet server (irc.efnet.org), the #xboxhacker channel is dedicated to discussing programming and hacking of the Xbox console.

PlayStation 2

Topics in this Chapter:

- **Introduction**
- **Commercial Hardware Hacking: Modchips**
- **Getting Inside the PS2**
- **Installing a Serial Port**
- **Booting Code from the Memory Card**
- **Other Hacks: Independent Hard Drives**
- **PS2 Technical Details**
- **Homebrew Game Development**
- **PS2 Resources on the Web**

**Chapter Note: Circumventing copy protection measures (for example, by using modchips) in order to play pirated or unauthorized games may be unlawful in your country. Both U.S. and other governments have successfully prosecuted sellers of modchips and similar devices. Please check with your country's laws regarding such activities.

Introduction

With over 70 million consoles sold worldwide as of April 2004, Sony's PlayStation 2 (PS2) has the largest user base of the current generation of gaming consoles. Surprisingly, the PS2 has the least amount of hardware hacks and homebrew software projects, compared to Microsoft's Xbox or Sega's now defunct Dreamcast. There is an active community of PS2 homebrew software developers, but its size doesn't compare to the number of Xbox or Dreamcast homebrew hackers. Outside of "modchip" manufacturers, very few hardware hackers are dedicated to exposing the PS2's internals.

One of the reasons that not a lot of hacking is done on the PS2 is that a large number of the talented individuals who decide to reverse-engineer the PS2's hardware are modchip manufacturers. These manufacturers guard the information they find as trade secrets, rarely revealing information to the public (usually providing just enough detail to allow a user to install the device). Another reason is that even though the PS2 contains a few standard connections such as USB and FireWire (IEEE 1394.1), internally the PS2 is vastly different from other "standard" architectures such as the PC architecture that the Xbox is based on. A lot more work is involved in locating and snooping the data buses and determining the signals output by the custom chips found on the PS2's mainboard.

Commercial Hardware Hacking: Modchips

For all consoles, software piracy is a thriving business. Although console manufacturers implement security features that make it difficult or impossible for the average gamer to copy games, hardware hackers employ seasoned techniques to make short work of defeating hardware security methods. Such techniques include snooping the system's data bus with a logic analyzer and dumping an image of the BIOS to look for software workarounds. Once just simple shims to allow a user to play unauthorized copies of games, modchips are now complex devices that allow much more.

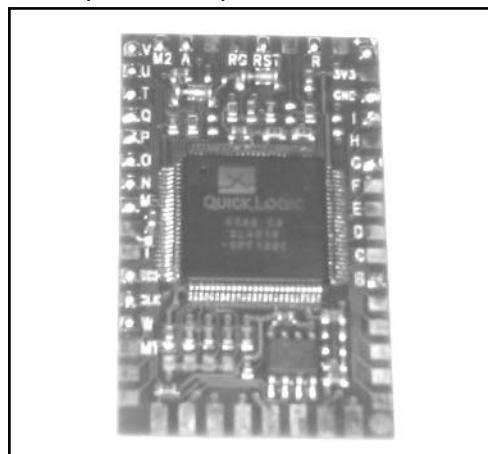
When a viable set of exploits and workarounds is found to bypass a particular security feature, the modchip is born. Modchips are small printed circuit boards (PCBs) with wires that attach to various components on the console's main board. A modchip is usually controlled by a Microchip PIC or standard programmable logic (such as a programmable logic device [PLD]), but modern modchips for the PS2 and other systems include an FPGA and Flash ROM so that they can be updated with bug fixes and new features. Normally, a modchip sends a signal to the console's security system to fool the system into thinking the user has inserted a valid game disc. Modchips can also bypass other systems, including code in the BIOS, or the default video mode (PAL or NTSC) output by the console's graphics hardware.

On the PS2, modchips are primarily used to bypass the hardware authentication performed on PS1 and PS2 game discs. They also are used to patch the BIOS to disable Macrovision for recording DVD content to a VCR, to enable region-free DVD playback, and to override the video mode used by games in other regions (for example, playing PAL games on an NTSC system). The original PS2 modchips were crude, simple devices that could only bypass PS1 game disc authentication and facilitate a "swap trick" for PS2 game discs only if the user had bought a separate and unsupported boot disc. These boot discs (written and sold by unlicensed third-party vendors) contain code to stop the PS2's DVD drive from spinning. Once the drive has spun down, the user forces the drive open, using

a special case modification (called a *fliptop*) or sometimes using a plastic knife or a credit card-sized device. The user then puts in the copied PS2 game and closes the drive, and the PS2's hardware authentication will be defeated, having failed to detect the forced opening and closing of the drive.

The main disadvantage of physical swapping is that using an object to force open the PS2's DVD drive can shorten the life of the drive. Modern modchips have done away with the destructive swap method by authenticating a copied PS2 game disc directly in hardware. Some of these modchips include an FPGA and Flash memory so that users can apply bug fixes and add new features by simply booting a disc. They also include a special loader that executes a program stored on the PS2 memory card when the PS2 is powered on. This program is usually another loader, but with a graphical user interface (GUI) for running other programs and utilities for installing programs to the memory card or a hard drive from a CD. Figure 4.1 shows the LisaZero PAL-only modchip, which uses a QuickLogic PLD that contains fixed boot code.

Figure 4.1 The LisaZero “No-Swap” Modchip



The engineers and hackers behind commercial modchips invest a lot of time and money reverse-engineering the hardware and testing their findings. The low-level information that they discover is rarely (if ever) shared with the public, since by doing so they would lose their edge over competing manufacturers. Other modchip makers can still offer the same or similar features by reverse engineering a competing modchip or using the same techniques that they would use to document the PS2 hardware platform. Although some of the information they discover is only useful to someone who wants to bypass PS2 hardware security, most of the information would benefit anyone who wanted to develop their own hardware or software for the PS2. This information includes the pinouts and signals of the PS2's processors and buses, expansion port, and BIOS.

Not having this information available to everyone stifles the PS2 hardware hacking and home-brew development communities. As presented later in this chapter, some of the hidden hardware information enables the PS2 developer to gain better control over the system and access to powerful debugging methods. My main motivation for releasing the Independence exploit (see the “Booting

Code from the Memory Card” section) was to allow anyone interested in writing PS2 software to do so without having to physically modify their console.

NEED TO KNOW...



Some sections of this chapter contain source code examples. Unless specified otherwise, all examples were written for use with PS2Lib, an Open Source library for the PS2. You can obtain PS2Lib from <http://ps2dev.sourceforge.net/ps2lib.html>.

A couple of comments on the conventions used in the code:

- PS2Lib’s *tamtypes.h* header defines the basic types used on the PS2. A *u* followed by a number indicates an unsigned type with the specified number of bits, and an *s* followed by a number indicates a signed type. For example, *s8* indicates an 8-bit signed integer and *u32* indicates a 32-bit unsigned integer.
- *tamtypes.h* also defines macros for convenient access to hardware registers. These macros are similar to the *inb()* and *outb()* style macros found in low-level PC programming. The *_lw()* macro is a synonym for the MIPS *lw* instruction, which returns a 32-bit value read from the given address (the address is specified as an unsigned 32-bit integer). Likewise, the *_sw()* macro stores a 32-bit value to the given address. Each of these macros represents the corresponding MIPS instruction, so there are *_lb()*/*_sb()*, *_lh()*/*_sh()*, and *_ld()*/*_sd()* macros for reading and writing 8-bit, 16-bit, and 64-bit values, respectively.

Getting Inside the PS2

The following is a guide to identifying your PS2’s mainboard revision and steps to disassemble the PS2.

Mainboard Revisions

As of August 2003, there are 11 known major revisions of the PS2 mainboard and about a dozen BIOS revisions. Mainboard revisions are usually denoted with *V* (for version), followed by the major revision number—for example, V7. The revision number is commonly referred to as just the PS2 version number. Numbering starts at V0, which is the version of the original PS2 launched in Japan. There is no notation for minor revisions.

There are a number of reasons that Sony would want to revise the PS2 internally:

- To fix hardware and software bugs
- To move separate peripherals onto a single chip to lower manufacturing costs
- To implement new security measures to keep attackers out

The problem with major mainboard revisions is that the physical layout of the board and its components changes from one revision to the next. This means that the instructions for locating a particular component or testpoint will be different on a V1 than on a V7. Some revisions have a close

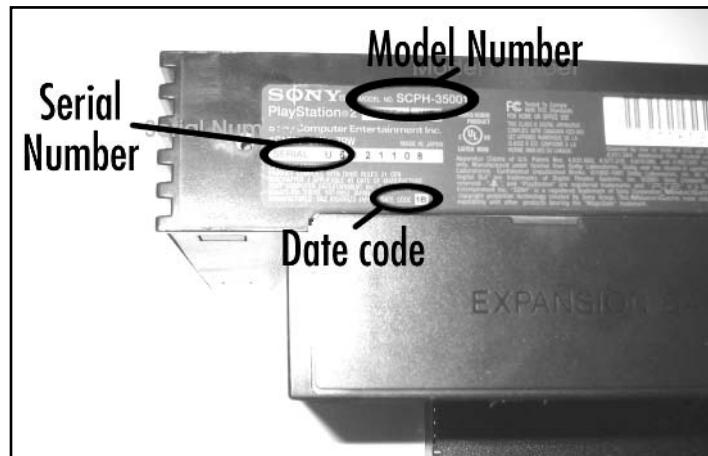
enough layout so that the same instructions will apply to both, as is the case for V5s and V6s. The hacks demonstrated in this chapter have been performed on a V4 PS2. If you own another revision, you will need to adapt the instructions to your mainboard. Where possible, I will note instances where the boards differ across revisions.

Identifying Your Mainboard

To find out your PS2 version number, do the following (see also Table 4.1):

- Flip over the PS2 and count the number of square screw covers on both the bottom and near the expansion (or PCMCIA) slot.
- Look at the sticker on the rear of the PS2. First, write down the text that follows “Model Number.” Next, record the first two digits of the serial number and finally, the two-digit date code (Figure 4.2).

Figure 4.2 Model and Serial Numbers and Date Code



- Check for any physical variations that would set your PS2 apart. For example, only V0 (Japanese) machines have a PCMCIA slot; the rest have an expansion bay for the internal HDD. V9 and higher are missing the IEEE1394 port (next to the USB ports) and may have a small infrared receiver between the Reset and Eject buttons.
- If your model number is SCPH-10000 or SCPH-15000, or if you have a rear PCMCIA slot instead of an expansion bay, you have a V0 PS2.
- If you have 10 screws on the bottom of your PS2, it's a V1, V2, or V3. Refer to Table 9.1 to determine the exact version.
- If you have eight screws on the bottom of your PS2, you have a V4 or higher. V4s include model numbers SCPH-30000 through SCPH-30006 and SCPH-35001 through SCPH-35006. If you have an electrical hazard warning underneath the PS2, it's a V4. If you remove

your expansion bay cover and notice a metal shield on the inside of the cover, you most likely have a V4.

- V5 or V6 PS2s (there aren't a lot of differences between these internally) use model numbers SCPH-30000R through SCPH-30006 R (note the space) or SCPH-30000 through SCPH-30004. To distinguish a V5 or V6 from a V4, remove the expansion bay cover and look for a small screw at the top of the expansion bay, closest to its left side. If this screw is there, you have a V5 or V6. Also check the inside of the expansion bay cover. If it's plastic, chances are you have a V5 or V6.
- If your model number includes SCPH-39000 through SCPH-39004 or SCPH-37000, you have a V7.
- If your PS2 is from Japan and your model number is SCPH-39000 or SCPH-39006, you might have a V8 PS2. I don't know of any major differences on the mainboard between V7s and V8s.
- If your model number includes SCPH-50000 through SCPH-50004 and your date code is not 3D, you have a V9 PS2. If your date code is 3D, you have a V10. V9s and V10s are also missing the FireWire port and may have an infrared receiver sandwiched between the Reset and Eject buttons.

Table 4.1 Identifying V1, V2, and V3 PS2s

Version	Serial Number (First Two Digits)	Date Code
V1	U1	0D
V2	U0	0D
V3	U1	1A
V3	U2	0D

Opening the PS2

Our first hack is performed on the bottom of the PS2's mainboard. Opening the PS2 and exposing the mainboard can be tricky if you've never done it before. The following instructions apply to V4 PS2s, so you may have to modify them slightly for other PS2 versions.

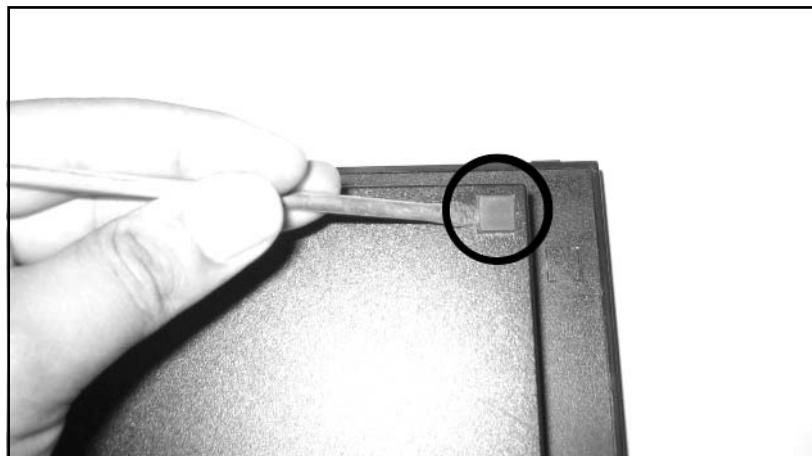
The only tools you will need for this hack are a Phillips screwdriver and a jeweler's size flathead screwdriver.

WARNING: HARDWARE HARM

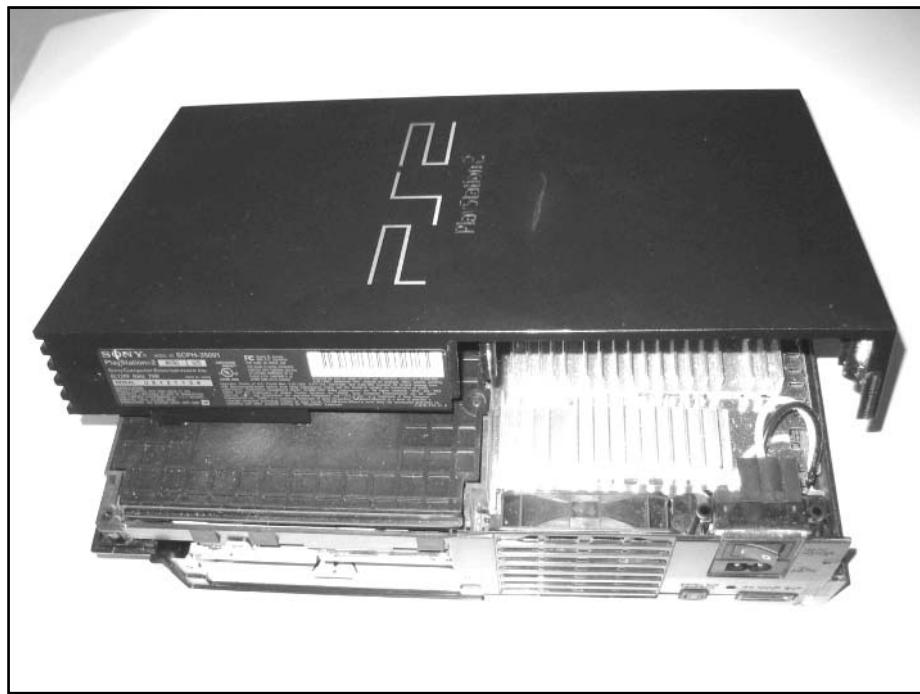
The PS2 mainboard and connected assemblies contain many electrostatic-sensitive components. You must always be grounded before touching the inside of your PS2 or you risk damaging these components. The easiest way to ground yourself is to buy an anti-static wrist strap and connect it to a grounded source. If you do not have a wrist strap, you can ground yourself by touching a piece of metal (such as the edge of your desk) before touching the inside of your PS2.

1. Turn the PS2 over so the bottom is facing up. You should notice either eight or 10 square indentations; these are covers for the case screws. Using either your fingernail or a small flat-head screwdriver, pry up each of these covers to expose the screws (see Figure 4.3).

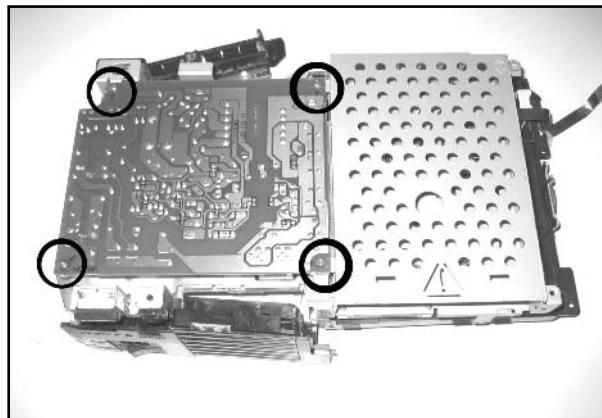
Figure 4.3 Removing the Plastic Screw Covers



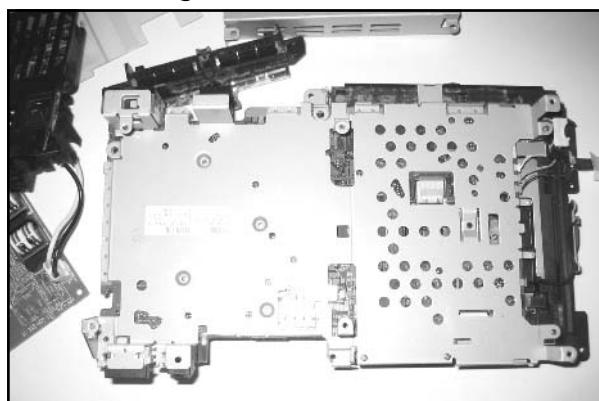
2. Using a Phillips screwdriver, remove the case screws. You may meet some resistance because the screws could be glued shut. Just turn them counter-clockwise until you hear a snap and the screw should easily turn the rest of the way.
3. If you have a warranty sticker (usually found next to the A/V connector on the back), remove it.
4. Flip the PS2 back over and face the front of the PS2 away from you. The expansion bay should be to your left and the A/V connector to your right. Slowly lift the top half of the case (see Figure 4.4). You will have to move it slightly forward to remove it from the joypad assembly and the DVD drive. Be careful not to lift the case too fast, since it is still connected to the Reset/Eject button assembly.

Figure 4.4 Removing the Top Case Cover

5. Remove the Reset/Eject button assembly from the top half of the case by pulling the assembly down from its corner until it snaps off. Pull back on the assembly and it should slide out of the notch in the front of the case. Set the assembly down next to the PS2, keeping the ribbon cable intact.
6. Remove the two brass screws that hold the joypad assembly in place.
7. Remove the brass screw on the far right side of the rear Fan/Power assembly. Do not remove the screw in the assembly that is closer to the DVD drive.
8. Carefully lift up the Fan/Power assembly and you should notice another brass screw between the optical connector and the A/V connector. The fan is connected to the mainboard, so if you lift up the assembly too fast you could damage the connector. Remove the screw there.
9. While holding the joypad assembly and the Fan/Power assembly, turn the PS2 over. Make sure the front of the PS2 is facing away from you. You should now be able to lift up the bottom of the PS2's case. Set it aside.
10. Your unit should now resemble the one in Figure 4.5. The green PCB on the left is the power supply unit (PSU). Remove the four brass screws that secure the PSU in place.

Figure 4.5 The Bottom of PS2 and PSU

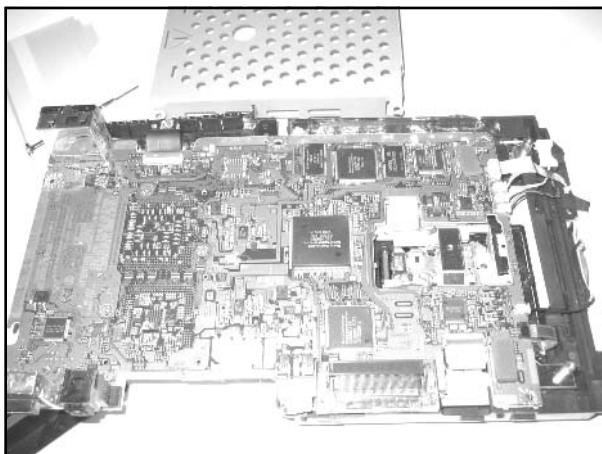
11. The PSU is also connected to the PS2 mainboard with a four-pronged connector. Carefully lift the PSU to disconnect it. You should notice a small two-wire connector going from the fan to the mainboard. Pull straight up on this connector, grabbing the wires that are the closest to the mainboard connector. Set the PSU and Fan/Power assembly aside.
12. Remove the plastic sheet on top of the metal casing, and remove the metal housing for the internal hard drive (see Figure 4.6).

Figure 4.6 Bottom Metal Casing

13. Remove the eight tiny brass screws. There are four of these screws underneath the plastic sheet. Two more screws secure the expansion bay connector, and two more are along the right side of the metal casing, above the DVD drive. Remove the tiny black screw located underneath the DVD drive.
14. The large metal casing is attached to the DVD drive with two tabs closest to you and one tab in the front. Pry these tabs and carefully lift up the metal casing.

15. To give you easier access to the mainboard, you might want to prop it up on the left side near the A/V connector. I used the expansion bay cover as a prop (see Figure 4.7).

Figure 4.7 Bottom of PS2 Mainboard Exposed



Your PS2 is now completely disassembled and we can begin the hacks! To reassemble the console, carefully follow the previous steps in reverse.

Installing a Serial Port

In embedded systems, the serial port is often the communications lifeline of the system. It can be used to load programs, receive status messages, and for debugging software running on that system. Like most System-on-a-Chip (SoC) designs, the Emotion Engine (EE) includes an on-chip Serial Input/Output (SIO) port used internally by the EE's kernel to output debugging and status messages and to start the kernel debugger. For additional information regarding the PS2 EE please refer to the “PS2 Technical Details” section in this chapter.

In your own homebrew software, SIOs can be used to output debugging messages, or it could be used to support a full remote debugger such as GDB (<http://sources.redhat.com/gdb>). The PlayStation 2 Linux operating system also supports a serial console over SIO. The main benefit of using SIO over a USB link cable or the official PS2 Network Adapter is that SIO is a direct connection to the EE, whereas the other methods are arbitrated by the I/O Processor (IOP). If something goes wrong and the IOP crashes, you have little to no means of recovering your program on the EE. Also, the SIO cable we're going to build supports up to 115.2kbps, which is a fairly high-speed connection.

The cable that we'll be building for this hack requires only five wires soldered onto the PS2's mainboard and a simple interface circuit that needs only 15 connection points.



Preparing for the Hack

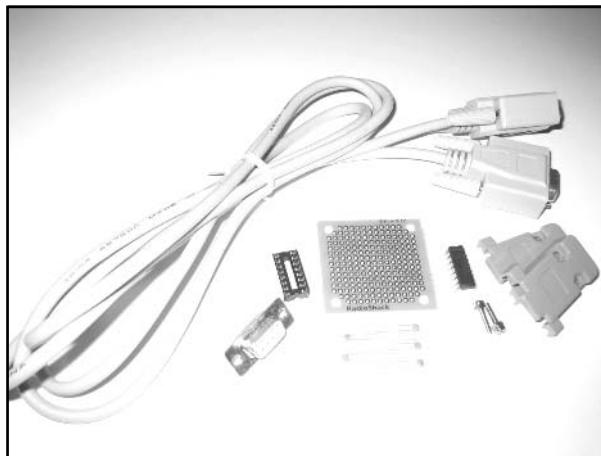
Table 4.2 lists the parts required to build the SIO cable and interface board. Figure 4.8 shows a picture of the components. You can order samples of Maxim's MAX3323EEPE from the company's Web site (www.maxim-ic.com). Be sure to order the DIP version of the chip. The rest of the parts can be found locally at Radio Shack or online at many electronics distributors.

Table 4.3 shows the wire colors I chose for the connections.

Table 4.2 Parts List for the SIO Cable

Quantity	Part	Comments
1	MAX3323EEPE	Maxim, www.maxim-ic.com
5	0.1µF monolithic capacitors	Radio Shack part #272-109
1	Female DB9 connector	
1	Plastic DB9 hood	
1	Sixteen-pin IC socket	Radio Shack part #276-1998
1	PC board	Radio Shack part #276-148
5	30AWG wire	Approximately 12" in length
1	DB9 serial cable	Optional
1	Five-pin plastic male and female connectors	Optional

Figure 4.8 Required Parts for the SIO Cable



The required tools for this hack are:

- Soldering iron and solder
- Wire Cutters

- Jeweler's size flathead screwdriver (optional)
- No-clean solder flux (optional)
- Masking tape, epoxy, or hot glue

Table 4.3 Serial Cable Wire Colors

Color	Signal
Red	+3.3V (Vcc)
Black	Ground (GND)
White	EE core voltage (VCORE)
Blue	EE_TXD and PC_TXD
Green	EE_RXD and PC_RXD

Performing the Hack

WARNING: HARDWARE HARM

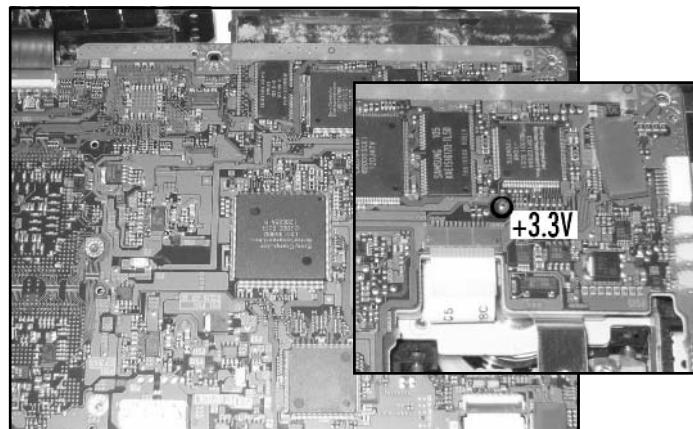


The PS2 mainboard contains many heat-sensitive surface-mount components. If possible, use a low-temperature soldering iron to avoid damaging the parts.

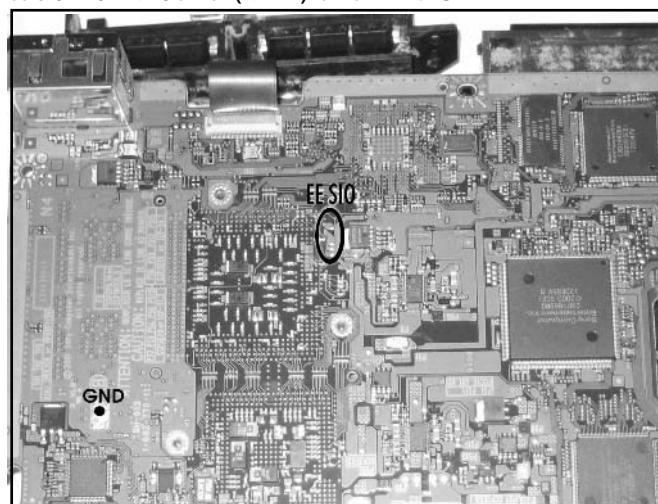
The bottom side of your PS2's mainboard should be exposed, with the A/V connector closest to you and the memory card connector facing away from you. See the “Getting Inside the PS2” section for details on how to get to your mainboard. If you have a V4 or higher mainboard, you should be looking at something similar to Figure 4.7.

Perform the following:

1. First, locate the +3.3V pad. Figure 4.9 shows the location on a V4 mainboard. Most mod-chip installation Web sites (such as www.dms3.com) show the +3.3V location for your specific mainboard version. Solder one end of the red wire to this point.

Figure 4.9 Location of the +3.3V (Vcc) Pad

2. On the left side of the mainboard, you should see printed text including “GH-” followed by a three-digit number. Somewhere in this area is a silver rectangle we’ll use for ground. Figure 4.10 shows this location. You may see oxidation (small brown spots) on the rectangle. Solder one end of the black wire to this point.
3. To the right of the text, you should see many small capacitors surrounding the area and several large ones positioned in the center. Look at the upper-right corner and you should notice four small silver square-shaped pads positioned in a vertical line. There will be a gap in between the top pad and the bottom three. This is the EE’s SIO port and is as denoted in Figure 4.10.

Figure 4.10 Location of Ground (GND) and EE SIO

- The first point that we'll connect from the EE SIO will be our most difficult one: the EE_RXD pin. The top square pad is EE_TXD. If you follow the EE_TXD pad back to its via, you'll find EE_RXD sitting on a via directly above it. Figure 4.11 shows a close-up of the EE_RXD via, EE_TXD pad, and VCORE capacitor. Be careful not to connect the incorrect via in the area which is attached to a small resistor; the EE_RXD via isn't connected to anything else on this side of the board. Carefully solder one end of the green wire to the EE_RXD via.

WARNING: HARDWARE HARM



A *via* is a small circle-shaped hole found on the mainboard. It usually connects one layer of the circuit board to another. In this hack, one of our points (EE_RXD) is connected to a small via. Note that vias are different from *pads*, which are small square- or circle-shaped solder points.

Because vias connect the different layers of the mainboard, if they are damaged they can "sink" into the mainboard layer and the mainboard can be destroyed. This happens if you apply too much heat to the via. To avoid damaging your mainboard when connecting EE_RXD, do the following:

- Using a screwdriver, gently scratch the surface of the via to remove the solder-mask that covers it.
- Apply paste flux to the exposed via.
- Apply a small amount of solder to your soldering iron and quickly touch it to the via so that the solder moves from the iron onto the via.
- Tin the wire used for connecting EE_RXD.
- Apply a small amount of flux to the tinned wire.
- With the tinned wire on top of the via, momentarily touch the iron to the glob of solder on the via so it attaches to the tinned wire. The key here is being as brief as possible when heating the via, but still ensure that there is a good solder connection between the wire and the via.

-
- Next, solder one end of the blue wire to the square EE_TXD pad, denoted in Figure 4.11.
 - The last point we need to solder to on the mainboard is +1.7V, also known as Vcore. This is usually found on one end of the large black or beige capacitors underneath the EE. Usually there is one right next to the second square pad as shown in Figure 4.11. Solder one end of the white wire to the end of the capacitor where you see a small beige dot on the mainboard.

Figure 4.11 The VCORE, EE_TXD, and EE_RXD

7. With all five points connected (see Figure 4.12), use either masking tape, epoxy, or hot glue to secure them to the mainboard to act as a stress relief. If you have V4 or higher, there should be a notch on the left side where the cables for the power button assembly fit. Route your wires through this notch so that they stick out of the left side of the PS2 (see Figure 4.13).

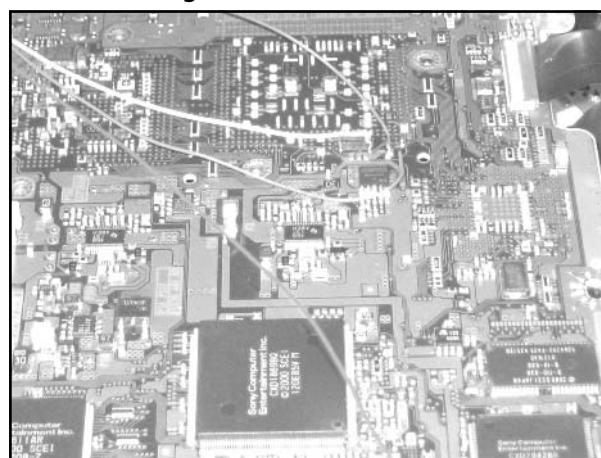
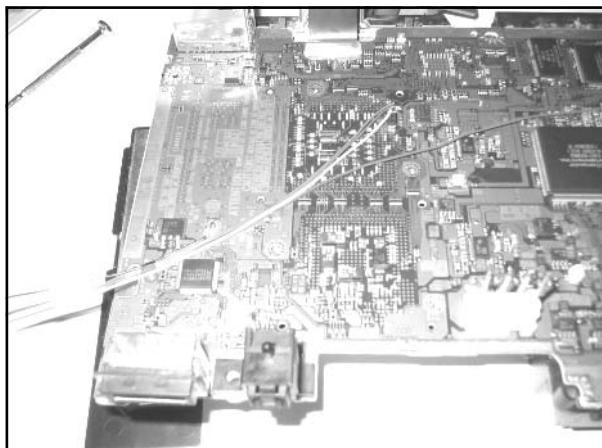
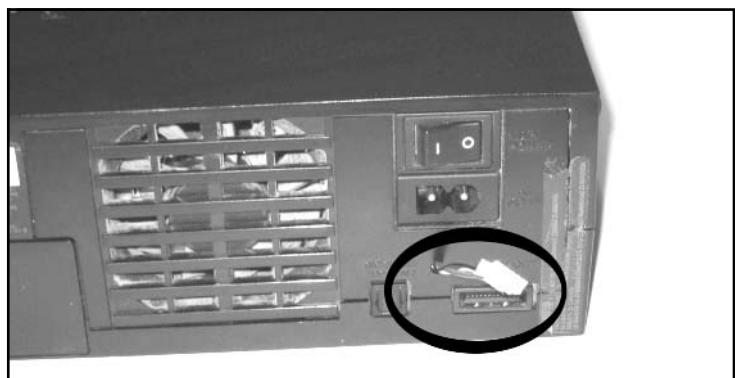
Figure 4.12 Installed SIO Wiring

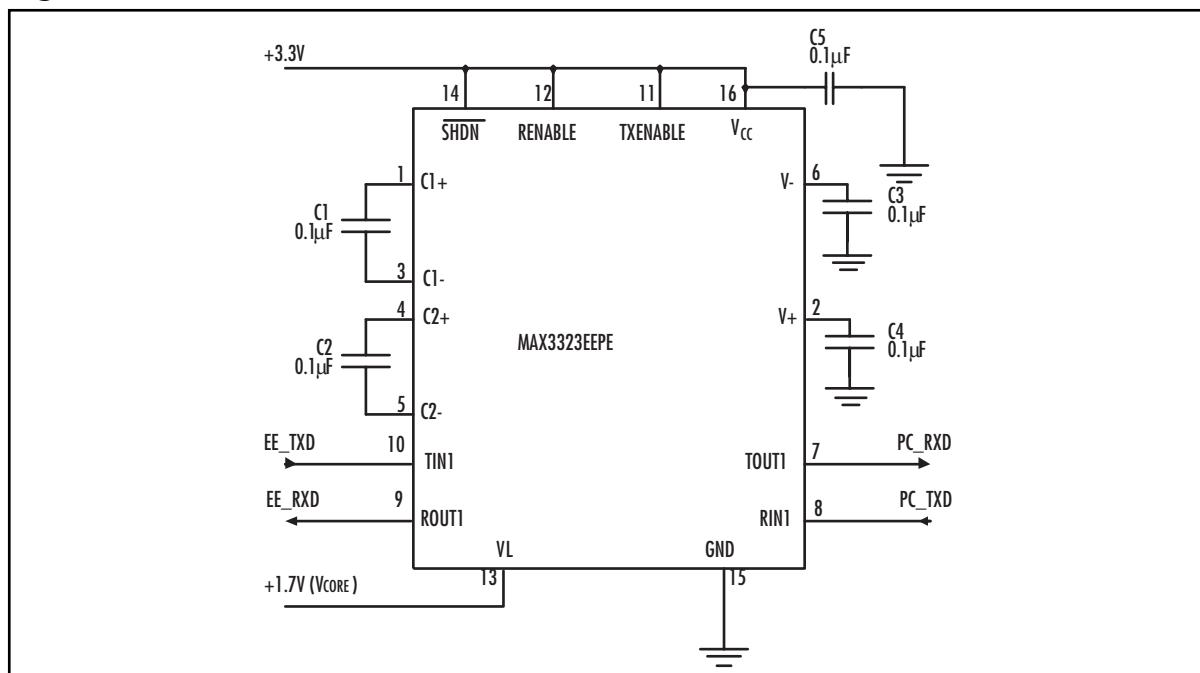
Figure 4.13 Mainboard Ready for Reassembly

8. Now it's time to reassemble your PS2. As you replace the metal casing that fits over the bottom, you should notice a small screw hole between the A/V connector and the optical connector. This hole is normally used to secure the PS2 to a store display. If you plan on mounting the interface circuit board on the outside of the PS2, you can feed your wires through here and through the fan and power button assembly. The wires can be seen coming out of this hole in Figure 4.14. To finish the PS2 reassembly, follow the previous "Getting Inside the PS2" section in reverse.

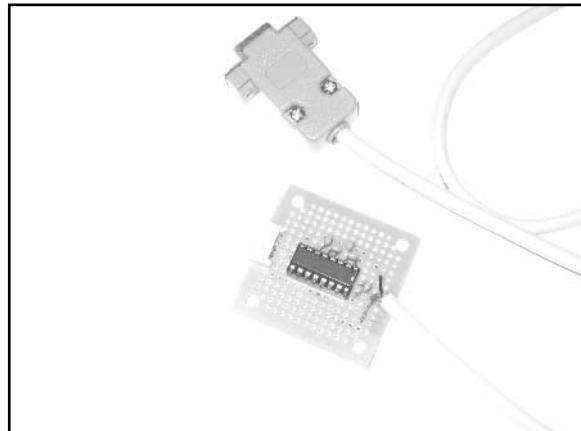
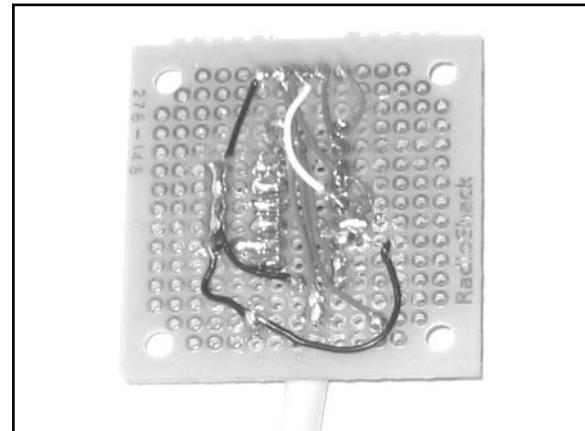
If you decide to use a five-pin plastic header to interface to your SIO board, you should connect the wires to the female end at this time. Figure 4.14 shows a 5-pin plastic header attached to the wires.

Figure 4.14 The Completed Five-Wire SIO Connector

With all of the internal wiring completed, it's time to build the external interface board. The schematic is shown in Figure 4.15. On the DB9 connector of the PC serial cable, PC_RXD will connect to pin 2, PC_TXD will connect to pin 3, and GND will connect to pin 5.

Figure 4.15 SIO Interface Schematic

Figures 4.16 and 4.17 show the top and bottom of the completed board.

Figure 4.16 Top View of External SIO Interface**Figure 4.17** Bottom View of External SIO Interface

Testing

To test your interface board, connect it between the PC and PS2. Using a terminal program (such as HyperTerminal which comes with Windows) set your serial port to 38400 baud, 8 data bits, no parity, 1 stop bit, and no hardware flow control. Power on your PS2. If the cable is working properly, you should see messages similar to the ones in Figure 4.18.

Figure 4.18 EE Boot Messages

```

SIU - HyperTerminal
File Edit View Cell Transfer Help
D S G C E F

# Initialize Scratch Pad ...
# Restart Done.
# Initialize Scratch Pad ...
# Initialize Done.

EE DEC12 Manager version 0.06 May 11 2000 18:08:48
CPUID=2e20, BoardID=ffff, ROMGEN=2000-1228, 32M

# Restart Without Memory Clear.
# Initialize GS ...
# Initialize INTC ...
# Initialize TIMER ...
# Initialize DMAC ...
# Initialize VU1 ...
# Initialize VIF1 ...
# Initialize GTF ...
# Initialize VU0 ...
# Initialize VIF0 ...
# Initialize IPU ...
# Initialize FPU ...
# Initialize Scratch Pad# Initialize Scratch Pad ...
# Restart Done.

Connected DUE229 Auto detect 38400N1 SCROLL CAPS NUM CapsLock PrintEcho

```

Under the Hood: How the Hack Works

This hack works by taking advantage of the undocumented SIO port of the PS2's Emotion Engine (EE). Details of the SIO can be found in the "PS2 Technical Details" section. A standard RS232 serial port in desktop PC's generally use +/-12V to define a logic 0 (space) and logic 1 (mark). The EE is powered using a +1.7V source, so if the SIO pins were connected directly to a PC serial port without any voltage conversion, the signals that the PC sent to the EE could possibly destroy the device. The MAX3323 device is a "level shifter" which converts the voltage levels output by the EE to the levels the PC requires and vice versa. The +3.3V power source supplied by the PS2 is used to power the MAX3323.

Booting Code from the Memory Card

On August 15, 2003, I released the "Independence exploit" for the PS2 that enables the PS2 to boot any software stored on a standard PS2 memory card simply by launching a PS1 game. Any PS1 game can be used as long as the game's unique *title ID* has been written to the file on the memory card that contains the exploit. When you power on the PS2 with a PS1 game already inserted into the DVD drive, the exploit takes over as soon as the PS2 displays the Sony Computer Entertainment banner. The exploit can also be triggered by inserting a PS1 disc and launching the disc manually from the PS2's browser.

NEED TO KNOW... PS2 INDEPENDENCE WEB SITE



The official Web site for the Independence exploit is www.0xd6.org/ps2-independence.html. Here you can find the latest version of the exploit and tutorial information about memory card save files and setting up the exploit using Windows. Since the exploit's release, numerous guides and tutorials have popped up on the Web detailing the steps to get various pieces of homebrew software to run from the memory card. Some of this software includes the Naplink USB boot loader (<http://naplink.napalm-x.com>), the Pukklink and ps2link Network Adapter loaders (www.ps2dev.org, in the Loaders section), and the PS2Reality group's MediaPlayer (www.ps2reality.net). Many more types of software are currently being developed, and using the exploit to load a boot loader has proven to be the easiest way to load custom software. Though the term "exploit" may sound nefarious, this discovery has opened the doors to an entire community of homebrew programmers for the PS2.



Preparing for the Hack

The most difficult part of this hack is finding a way to get the files onto a memory card. If your PS2 is already modded or you use the "swap trick" to boot CDRs, you can use a disc image called Exploit Installer written by Nicholas Van Veen. A tutorial for this installation is available <http://ps2emu.netfirms.com/ps2homebrew3.shtml>.

NEED TO KNOW... THE RIGHT MEMORY CARD



The Independence exploit will only work with a Sony PS2 memory card or any third-party memory card that supports MagicGate. It will not work with a PS1 memory card or with non-MagicGate PS2 cards.

If you have no other method of loading a program on the PS2, you can buy one of several memory card adapters that connect to the PC via a USB cable. Doing a Web search for "USB memory adapter" or "PS2 memory card adapter" will give you a bunch of places to buy one, including www.liksang.com.

Performing the Hack

Preparing TITLE.DB

In order for the exploit to "trigger" off of your PS1 game, you must add that game's unique title ID to the TITLE.DB file that is stored on the memory card. I've written a command-line utility called *titleman* to make it easier to do this. You can grab a Windows executable version of *titleman* from the tutorial section of the Independence Web site, or you can choose to compile it from source yourself.

NEED TO KNOW...



I don't know of any graphical utilities to manage the TITLE.DB file. You need to have some basic familiarity with the text-based command line, which is sometimes called Command Prompt in the Windows Start menu. It can be accessed by using **Start | Run** and typing in either **command** or **cmd** and pressing **Enter**. If you are using a UNIX-based system (including Mac OS X), consult your system's manual to find out how to access and use the command-line.

The titleman program options are listed in Table 4.4. The first step in creating a TITLE.DB file is to use the **-c** option. This creates TITLE.DB on disk and stores the exploit payload and a few standard title ID entries.

Table 4.4 The titleman Program Options

Option	Description
-c	Create a new TITLE.DB and initialize it
-a	Add one or more title IDs to TITLE.DB
-d	Delete one or more title IDs from TITLE.DB
-l	List all of the title IDs stored in TITLE.DB
-o	Specify an alternate output file
-v	Be verbose; output status messages

The original titleman tool contains the PS2 executable that launches the BOOT.ELF file from the memory card. BOOT.ELF is stored in the Your System Configuration folder that can be seen in the PS2's browser. The physical name of this folder is:

- **BADATA-SYSTEM** for PS2s bought in North America
- **BEDATA-SYSTEM** for PS2s bought in Europe
- **BIDATA-SYSTEM** for PS2s bought in Asia

When you are preparing the game save (see the section "Saving TITLE.DB to the Memory Card"), be sure to use the correct name for your PS2's territory.

Next, you must add the title ID for each PS1 game that you want to use to trigger the exploit. The title ID can usually be found underneath the game's ESRB rating icon on the printed side of the game disc. If you got the ID from the front of the disc, you need to convert it to the format that the PS2 expects. For example, on the front of my Street Fighter Alpha disc is the ID "SLUS-00197." To convert it, change the dash to an underscore and add a period between the third and fourth digits of the five-digit number following the dash. So, SLUS-00197 becomes SLUS_001.97.

You can also locate the title ID by loading the disc on a PC and opening the SYSTEM.CNF text file stored on the disc. The value after the equals sign (=) for the **BOOT** option is the title ID. If you

found the title ID in the SYSTEM.CNF file, it is already in the required format and does not need to be converted.

Once you have collected the title IDs for all the PS1 games you want to use, you can add them to TITLE.DB individually or by using titleman's *batch mode*. To add them individually, use **titleman -a title_id**. So in my example, I would type **titleman -a SLUS_001.97**.

Batch mode is an easier way to add multiple IDs at once. To use batch mode, place all your title IDs on individual lines in a text file, and use the **-a** option with the batch filename. You must prefix the filename with the “at” symbol (@) in order for it to be recognized as a batch file. You can include comments in the batch file by using a semicolon as the first character of the comment line. Figure 4.19 shows a sample batch file.

Figure 4.19 A Sample titleman Batch File

```
; Xenogears (disc 1)
SLUS_006.64

; Xenogears (disc 2)
SLUS_006.69

; Broken Helix
SLUS_002.89

; Suikoden
SLUS_002.92

; Suikoden II
SLUS_009.58

; Sentient
SCUS_941.10

; Blood Omen: Legacy of Kain
SLUS_000.27
```

The full listing of the steps required to add a single title (Street Fighter Alpha) are given in Figure 4.20.

Figure 4.20 Creating and Adding an Entry to TITLE.DB

```
$ titleman -c  
$ titleman -a SLUS_001.97
```

If you make a mistake while adding a title ID or if you want to remove an ID you've already added, use titleman's *-d* option. You can delete individual IDs the same way you added them, or you can use batch mode.

If you need to check which IDs you've already added to TITLE.DB, use the *-l* option.

Choosing BOOT.ELF

Version 0.1 of the exploit loads the executable file BOOT.ELF from the memory card after the exploit has taken over. BOOT.ELF is stored in the same folder as TITLE.DB. If you want to start developing your own software for the PS2, the best choice for BOOT.ELF is ps2link, an Open Source boot loader for loading programs via the official PS2 Network Adapter. The latest version of ps2link is available at www.thethirdcreation.net/tools, or from www.ps2dev.org (in the Loaders section).

If you are interested in streaming movies, MP3s, and Ogg Vorbis files over the network, you will want to install PS2Reality's MediaPlayer. You'll find tutorials for using the MediaPlayer with the exploit at www.ps2reality.net. If you want to fire up some of your old Sega Genesis games, Nicholas Van Veen's PGEN Genesis emulator also works well with the exploit. You can download PGEN from <http://pgen.gamebase.ca>.

To use any of the previously mentioned programs with the exploit, they must be setup properly so that they are loaded as soon as the exploit takes over. Some programs contain specific instructions in their included README file. Failing that, you can try renaming the program's main .ELF to BOOT.ELF (for example, rename PGEN_11.ELF to BOOT.ELF).

Saving TITLE.DB to the Memory Card

Now that we've created TITLE.DB listing all of the game titles that we want to trigger the exploit with, we need to find a way to get the file onto the memory card. There are several ways of doing this:

- **Nicholas Van Veen's Exploit Installer** This method either requires your PS2 to have a modchip installed or you must be willing to perform the swap trick. Although this method is convenient, I don't recommend swapping, since it is potentially damaging to your PS2 (or requires a hardware modification).
- **Using a commercial USB memory card adapter** Memory card adapters allow you to transfer "game saves" directly to a memory card inserted into the adapter and connected to your PC. To use this method, you must either create a brand-new game save using the name "Your System Configuration" and the save folder for your PS2's territory, or you can open an existing save and rename it. Once you have created your save, copy TITLE.DB and BOOT.ELF into it. Make sure that both the TITLE.DB and BOOT.ELF filenames are in all capitals or the exploit won't work. Once you have created this save, transfer it to your memory card using the software provided with your adapter.

- **The nPort utility** Napalm, the development team that wrote the Naplink USB boot loader, also developed a utility called nPort for transferring game saves between the PC and PS2 using an existing USB connection (in Naplink) or by using the official network adapter (in Pukklink or ps2link). The PS2 Independence Web site provides an .npo file (the format expected by nPort) that contains all the required export files. The site also features a tutorial that provides details on how to use nPort to save the exploit. You can find nPort at <http://wire.napalm-x.com>.

Independence!

Once you've saved the exploit to the memory card, insert the card into the PS2 and pop in any one of your PS1 games that you've added to the title ID list. After the disc loads, you should see a brief white flash followed by the initial screen of your BOOT.ELF program. If the game you insert goes into the normal PS1 emulator, recheck your TITLE.DB contents using the *-l* option and make sure you spelled the title ID correctly. Also verify that you have the correct PS1 game inserted. If you get a red screen while loading the exploit, make sure that you have BOOT.ELF in the same folder as TITLE.DB. Any other errors you notice will be specific to the application that you chose as BOOT.ELF.

Under the Hood: How the Hack Works

The PS2 is able to emulate PS1 software through a combination of hardware emulation and a software PS1 graphics emulator called PS1DRV. Whenever you boot a PS1 game on your PS2, the system browser first loads and executes PS1DRV from the BIOS. PS1DRV performs a number of configuration steps such as setting the disc speed and reading specific graphic parameters for the selected game. Finally, it initializes the graphics emulator and reboots the IOP into PS1 mode. Now the IOP takes over, and the PS1 game is loaded from disc. Graphics are emulated using a special Sub-CPU Interface (SIF) DMA channel from the IOP to PS1DRV on the EE.

When a PS1 disc is inserted into the PS2, the system browser reads the disc's title ID from a file on disc called SYSTEM.CNF. SYSTEM.CNF also contains other PS1 boot parameters such as the default video mode that the game was written to use. The game's title ID is passed along to PS1DRV so that PS1DRV can select parameters for its graphics emulator tuned to each individual game. If the system browser can't find SYSTEM.CNF on the PS1 disc, the value '????' is passed to PS1DRV instead.

When PS1DRV looks for game-specific parameters for its graphics emulator, it searches three locations: a built-in table, SYSTEM.CNF on disc, and a file on a memory card called TITLE.DB. The TITLE.DB file is stored in a system folder reserved for BIOS programs. If the PS2's region of sale is Asia, this folder is called BIDATA-SYSTEM; if the region is Europe, it's called BEDATA-SYSTEM; and if the region is North America, it's called BADATA-SYSTEM.

I found the Independence exploit within the routines that parse the title ID out of TITLE.DB. A main TITLE.DB loading routine that I'll call *load_mc_title.db()* is responsible for loading the TITLE.DB from the memory card into RAM. It calls another routine, *find_title_params()*, to locate the title ID within the loaded TITLE.DB and return a string value containing its parameter values. It's interesting to note that the way TITLE.DB is loaded into RAM makes the exploit extremely easy to

implement. TITLE.DB is loaded to the fixed RAM address 0x20800000, and the entire contents of the file are loaded. This means that we can include an entire program to take over after the exploit inside TITLE.DB and know exactly where it loads!

The *find_title_params()* routine is passed three parameters: the address where TITLE.DB is loaded (*title_db*), the address of a string variable to receive the parameter values (*params*), and the title ID to search for (*title_name*). It executes a loop to search each line (a line is terminated by an ASCII line feed, carriage return, or both) for the title ID. If it finds the title ID, it scans the rest of the line following the title ID to look for the terminating characters. Once those are found, it copies the data in between into the *params* string value. This copying is where the exploit occurs.

Ideally the parameter value associated with a title ID would be around 25 bytes, including the terminating line feed. In the *load_mc_title_db()* routine, Sony allocates 256 bytes to store this value, and these bytes are stored in RAM next to a very important EE register, the return address or *\$ra* register. On the MIPS architecture, when a routine executes another routine, it saves *\$ra* to RAM because the CPU automatically updates *\$ra* to point to the last instruction of the calling routine. Once the called routine has finished executing, *\$ra* still points to this instruction, so when the caller itself prepares to exit, it must first restore *\$ra* from RAM. In the *load_mc_title_db()* routine, the *\$ra* that is saved before calling *find_title_params()* is located after the space allocated to store the *params* string value.

When *find_title_params()* copies the parameter string into *params*, it uses the C function *strcpy()*, which copies one string of arbitrary length over another string. The *strcpy()* function has no boundary checking, so it will copy the entire string until it finds the terminating NUL (ASCII 0) character. This means that if we were to construct a string inside TITLE.DB that is longer than the 256 bytes allocated for *params*, we could overwrite the saved *\$ra* register (since it is stored in RAM after the *params* value). Whatever value that *\$ra* is overwritten with becomes the next address executed after *load_mc_title_db()* exits, and this can be any RAM address that is accessible by the EE.

This type of exploit, called a *buffer overflow*, is commonly found in software for which there is no boundary checking on strings or other values loaded from data files. It is easily preventable, as is the case with the Independence exploit in PS1DRV. Using the standard C function *strncpy()*, you can enforce the maximum length of the copied string. If Sony had originally used *strncpy()* with a maximum length of 256, the exploit would've never been possible.

So once we construct this string inside TITLE.DB, where do we point *\$ra*? Remember that *load_mc_title_db()* loads the entire contents of TITLE.DB into RAM at the fixed address 0x20800000. We can point *\$ra* to any address after TITLE.DB's load address. In the Independence exploit, I used the fixed address 0x20810110, to allow enough room for approximately 200 entries in TITLE.DB. When *load_mc_title_db()* exits, *\$ra* points to this address, and my code assumes control of the PS2.

Other Hacks: Independent Hard Drives

With the launch of Sony's PlayStation 2 Linux Kit (released in May 2002) and the official Network Adapter (released in August 2002), PS2s have gained an often overlooked addition: hard disk drive (HDD) support. In addition to the official Sony HDD that shipped with the Linux Kit, PS2 owners can also use off-the-shelf HDDs that fit the IDE connectors on the network adapter.

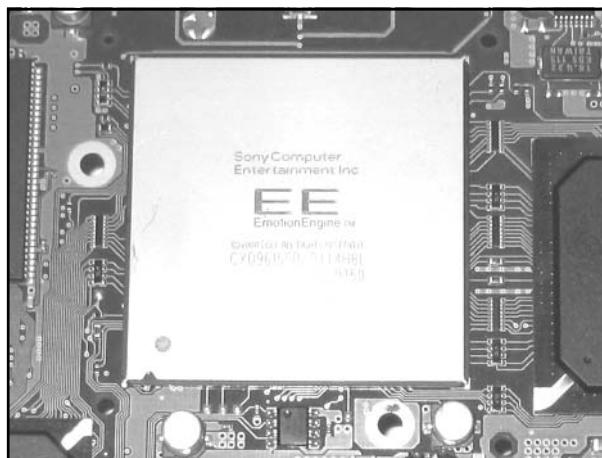
In November 2003, Nicholas Van Veen, along with others including myself, developed and released *libHDD*, a set of libraries and drivers for homebrew software to access the HDD. *libHDD* also includes support for Sony's official file system layout to be used by games that access the HDD. *libHDD* is available for download at <http://ps2dev.org/kb.x?T=967>. Using *libHDD*, programs designed to run from the memory card can access data or even other programs stored on the HDD. This opens up a range of potential projects, from emulators that support loading ROMs from the HDD (PGEN already does) to media players that play audio and video saved to the HDD. Hopefully, as more homebrew developers release HDD-aware programs, the PS2 will gain more recognition as a consumer hardware hacking platform.

PS2 Technical Details

The PS2 is built on a parallel architecture—it achieves high performance by splitting program tasks among multiple processors and coprocessors. A typical PS2 game will decide logic and game flow on the main processor, handle user input and audio on an auxiliary (or sub) processor, and transform 3D geometry on one of two high-speed coprocessors. Contrast this to the traditional PC-based architecture, where a single processor is dedicated to user input, game logic, and graphics output. Of course, modern 3D graphics cards now come equipped with a high-speed programmable graphics processing unit (GPU) that can offload 3D lighting and vertex calculations from the main CPU onto the graphics card.

Understanding the Emotion Engine

The heart of the PS2 is a 64-bit Toshiba MIPS processor dubbed the Emotion Engine, or EE (see Figure 4.21). The EE is the predecessor of ArTile Micro's TX79 line of System-on-a-Chip (SoC) processors. A SoC design is one in which all of the external peripherals required to manage the system are integrated onto a single chip. For example, a common TX79 chip, the TMPR7901, integrates the MIPS CPU core, an SDRAM memory controller, a PCI bus controller, and an Ethernet controller, among other peripherals. The main benefits of SoC processors are reduced manufacturing costs and better performance between the integrated peripherals. SoC designs usually contain one or more high-speed internal buses that interface with external peripherals using a slower, shared system bus.

Figure 4.21 The Emotion Engine

The EE includes the following documented on-chip peripherals:

- MIPS R5900 CPU core
- Two vector processing units (VUs or VPUs)
- Floating point unit (FPU)
- DMA controller (DMAC)
- Interrupt controller (INTC)
- Programmable timers
- Sub-CPU interface (SIF)
- Two VU interfaces (VIFs)
- Graphics Synthesizer (GS) interface (GIF)
- Image processing unit (IPU)

It also includes the following undocumented peripherals:

- RDRAM controller (RDRAMC)
- Serial I/O port (SIO; UART)
- JTAG (IEEE 1149.1) Boundary-Scan Interface

NEED TO KNOW... DIVINING DOCUMENTATION



A detailed examination of all of the EE's integrated peripherals would fill several chapters. The hardware manuals shipped on Disc 1 of the PlayStation 2 Linux Kit (www.playstation2-linux.com) provide an invaluable resource to the inner workings of the PS2's Emotion Engine and Graphics Synthesizer. If you do not own the kit, you can find basic information on the Emotion Engine's CPU core in the hardware manuals for the EE's descendant, the TX79 processor, at www.semicon.toshiba.co.jp/eng/index.html.

The Serial I/O Port

The SIO implements a high-speed UART with an 8-byte transmit first in, first out (FIFO) and 16-byte receive FIFO. It also includes the common CTS and RTS signals for hardware-based flow control. Although the SIO-related pins are unconnected in consumer PS2s, the EE's BIOS and runtime kernel use the SIO to output status messages during the PS2 boot process.

The SIO isn't documented in the EE User's Manual, so I had a look in the PS2's BIOS for the initialization and character output code. I also found a wealth of information on the SIO interrupt and hardware registers in the Toshiba TX79 core Architecture Manual, the TMPR7901 hardware manual, and the TMPR4925 hardware manual. It turns out that the TX79 Core Architecture Manual is nearly identical to Sony's EE Core User's Manual, except that all SIO-related documentation was removed from the latter.

After studying the BIOS and kernel SIO code, reading the available documentation, and writing a few test programs, I was able to assemble a reasonably accurate list of registers and definitions. The EE's SIO shares most of its registers with the TX79 core implementation, but I couldn't find any indication that DMA was supported. It also shares a few register definitions with the TX7901's UARTs. Table 4.5 shows the SIO register map.

Table 4.5 The SIO Register Map

Address	Name	Description
0x1000f100	SIO_LCR	Line control register
0x1000f110	SIO_LSR	Line status register
0x1000f120	SIO_IER	Interrupt enable register
0x1000f130	SIO_ISR	Interrupt status register
0x1000f140	SIO_FCR	FIFO control register
0x1000f150	SIO_BGR	Baud rate control register
0x1000f180	SIO_TXFIFO	Transmit FIFO register
0x1000f1c0	SIO_RXFIFO	Receive FIFO register

Using the TX79 manual, I found that when the SIO needs to interrupt the CPU, it triggers a Debug exception and sets bit 12 of the COP0 cause register. The CPU then decodes the SIO inter-

rupt status register to determine the cause of the interrupt. The EE kernel uses the SIO exception as a means to start the kernel's built-in debugger.

To initialize the SIO, you first write a value to SIO_LCR indicating the number of data bits and stop bits and whether to enable parity checking. You can also specify where the baud rate generator will get the clock source used to determine the baud rate. The next step is to reset both FIFOs and to optionally enable any interrupts. Finally, you need to calculate the divisor and the clock value used to maintain the baud rate. The code in Figure 4.22 is an example of how to initialize the SIO with the specified baud rate, using the standard transmission parameters of 8N1 (8 data bits, no parity checking, 1 stop bit).

Figure 4.22 SIO Initialization Code Example

```
#define SIO_FCR_FRSTE 0x01      /* FIFO Reset Enable. */
#define SIO_FCR_RFRST 0x02      /* RX FIFO Reset. */
#define SIO_FCR_TFRST 0x04      /* TX FIFO Reset. */

#define CPUCLK 294912000 /* Used to determine the baud divide value. */

void sio_init(u32 baudrate)
{
    u32 brd;                  /* Baud rate divisor. */
    u8 bclk = 0;               /* Baud rate generator clock. */

    /* 8 data bits, 1 stop bit, no parity checking, and use the CPU
       clock for determining the baud rate. */
    _sw((1<<5), SIO_LCR);

    /* Disable all interrupts. */
    _sw(0, SIO_IER);

    /* Reset the FIFOs. */
    _sw(SIO_FCR_FRSTE|SIO_FCR_RFRST|SIO_FCR_TFRST, SIO_FCR);
    /* Enable them. */
    _sw(0, SIO_FCR);

    brd = CPUCLK / (baudrate * 256);

    while ((brd >= 256) && (++bclk < 4))
        brd /= 4;
}
```

Continued

Figure 4.22 SIO Initialization Code Example

```
_sw((bclk << 8) | brd, SIO_BGR);
}
```

Character input and output are straightforward: You write to SIO_TXFIFO to output a character and read from SIO_RXFIFO on input. You'll need to check SIO_ISR to make sure that either the TX FIFO has room for another character or that the RX FIFO has at least one character ready to be read. The code in Figure 4.23, modeled after the ANSI Standard C Library *putc()* and *getc()* functions, demonstrates how to do this.

Figure 4.23 SIO Input/Output Code Example

```
int sio_putc(int c)
{
    /* Block until we're ready to transmit. */
    while (_lw(SIO_ISR) & 0xf000) == 0x8000;

    _sb(c, SIO_TXFIFO);
    return c;
}

int sio_getc()
{
    /* Do we have something in the RX FIFO? */
    if (_lw(SIO_ISR) & 0xf00)
        return _lb(SIO_RXFIFO);

    /* Return EOF. */
    return -1;
}
```

The I/O Processor

The I/O processor, or IOP, manages most of the on-board and external peripherals including memory cards, the sound processing unit (SPU), controller pads, and the DVD drive. It is a SoC design from LSI Logic based on the original PlayStation (PS1), and it includes all the PS1's main functions on a single chip. The core of the IOP is a MIPS R3000A CPU running at 36.864MHz (native speed). The IOP's native speed is approximately 1/8th the speed of the EE, which runs at

294.9Mhz. When emulating PS1 hardware, the IOP runs at the PS1's original speed of 33MHz. The IOP can directly access 2MB of RAM. It communicates with internal devices and external peripherals via the SBUS.

The Sub-CPU Interface

The IOP is also referred to as the sub-CPU, with the EE being the main CPU. The Sub-CPU Interface, or SIF, is a high-speed DMA connection between the IOP and EE. Using the SIF, each processor can transfer data directly into the other processor's RAM. The most common use of the SIF is a Remote Procedure Call (RPC) interface that allows the EE to call routines on the IOP. These routines access the low-level hardware driver corresponding to the RPC and return data via the SIF back to the EE. In this way, the EE can request to read from a file on an inserted DVD, continue its processing, and be interrupted by the IOP after the read request is complete. Also, using the SIF, the IOP can schedule controller data to be sent to a buffer on the EE on every Vblank ("vertical blank", 1/60th of a second for NTSC consoles, or approximately one frame of video). The EE can access this data without having to make an explicit request for it every single frame.

Homebrew Game Development

Throughout this chapter, we have discussed methods to load your own code onto the PS2. The PS2 homebrew development community is quite large and there are many resources available for aspiring programmers. The ones here should serve as a good starting point:

- **wiRe's frame:** <http://wire.napalm-x.com> Providing tools and projects for PS2 homebrew game development, including assemblers, sample games and code, and other miscellaneous goodies.
- **Official ps2dev home:** www.ps2dev.org Contains tutorials, code, boot loaders, and many other resources to get you started writing homebrew software for the PS2.
- **The Third Creation:** www.thethirdcreation.net The home of a monthly PS2 demo competition. I'd highly recommend that you download any demos authored by adresd; these are among the best.
- **Dan Peori's Web site:** www.oopo.net/consoledev Code and tutorials for all aspects of the PS2.
- **Lukasz Brunn's Mouthshut:** www.mouthshut.net The home of libITO, one of the first graphics libraries written for PS2 homebrew.

PS2 Resources on the Web

- **PlayStation 2 Linux Kit:** www.playstation2-linux.com Sony's official home of the PlayStation 2 Linux Kit, a hardware and software development kit that allows you to run Linux on the PS2.
- **PS2Emu:** www.ps2emu.netfirms.com Comprehensive news and information site for all things related to PS2 emulation and hardware hacking.
- **PS2 Scene Forums:** www.ps2-scene.org Active discussion forum for PS2 hacking and modifications. The site also contains a number of tutorials, reviews, and links.

Handheld Game Platforms



Nintendo Game Boy Advance

Topics in this Chapter:

- **Introduction**
- **A Very Brief History of Nintendo**
- **Opening the GBA Console**
- **Replacing the Display Lens**
- **Light Up Your LCD with the GBA Afterburner Mod**
- **Enhancing Your Afterburner with the GBA Stealth Dimmer Chip**
- **Nintendo GBA Technical Specifications**
- **Homebrew Game Development**
- **Other Hacks**
- **Nintendo GBA Resources on the Web**

Introduction

The Nintendo Game Boy Advance (GBA) is a sophisticated handheld video game machine containing a 32-bit microprocessor, 16-bit graphics capabilities, and stereo digital sound, though it's small enough to fit in your pocket. Millions of Game Boy Advance units have been sold, with upward of 100,000 units a week shipping around the world during peak months. And with more than 140 million units sold in all, it is the highest-selling video game system in history. Fourteen years, four models, and over 600 games later, Game Boy is the most successful video game franchise in history, outselling all others by a wide margin.

The Game Boy Advance has a long and fruitful history that goes back to 1989, when the original Game Boy was introduced. The Game Boy Advance is sort of the great-grandchild of that first Game Boy because it is the fourth Game Boy model. New as it is, however, the Game Boy Advance has now been supplanted by the Game Boy Advance SP (and the soon-to-be-released Game Boy DS, which has a dramatic shift in physical appearance and has two separate LCD screens). Granted, the internal hardware of the GBA and GBA SP is architecturally the same, but the SP model has some considerable new options, including a backlit screen. Table 5.1 shows an overview of the Game Boy models and their specifications.

Table 5.1 Game Boy Specifications

Model	CPU	Total RAM (KB)	Display Resolution	Colors
Game Boy	8-bit Z80, 4.17 MHz	8	160 x 144 pixels	4 (grayscale)
GB Pocket	8-bit Z80, 4.17 MHz	8	160 x 144 pixels	4 (grayscale)
GB Color	8-bit Z80, 8.0 MHz	48	160 x 144 pixels	56
GB Advance	32-bit ARM7, 16.7 MHz	384	240 x 160 pixels	32,768
GBA SP	32-bit ARM7, 16.7 MHz	384	240 x 160 pixels	32,768

Game Boy, 1989

The original Nintendo Game Boy (see Figure 5.1) was released in 1989, only four years after the Nintendo NES came out in the United States (see Chapter 7 for hacks of the Nintendo NES console). This first Game Boy operated on four AA batteries and was equipped with a Zilog Z80 microprocessor—the same processor used on many electronic devices in the 1980s. In fact, all the Game Boy models up to and including Game Boy Color featured a Z80 CPU, although later models were faster. The Game Boy's CPU runs at 4.17 MHz, which is comparable to the first IBM PC, which ran at 4.77 MHz.

Figure 5.1 Nintendo Game Boy, 1989



Game Boy Pocket, 1996

The Game Boy Pocket was a slimmed-down version of the Game Boy. Although the hardware specifications were essentially the same, the Game Boy Pocket required less voltage to operate (3 volts instead of 6) and thus could be powered by only two AAA batteries instead of four, slightly larger AA batteries.

Game Boy Color, 1998

The Game Boy Color (see Figure 5.2) was significantly more capable than the previous two models and greatly aided game developers with a faster CPU and more memory while still retaining compatibility with the older game cartridges. The Game Boy Color only requires two AA batteries and is therefore much lighter than the original Game Boy.

Figure 5.2 Nintendo Game Boy Color, 1998

The Game Boy Color is not only capable of displaying 56 colors on the screen at one time, but also enhanced existing Game Boy games to 32 colors, which greatly improved their appearance and playability. Original grayscale Game Boy games came to life on the Game Boy Color thanks to multiple shades of color. Obviously, backward compatibility was an important factor for Nintendo, primarily for marketing reasons. Being able to claim that its just-released console (the Game Boy Color) has a game library of several hundred titles is a marketing boon. Of course, a large percentage of those games are low-quality Game Boy titles. However, many excellent titles were released specifically for Game Boy Color, including the phenomenally successful Super Mario Bros. DX and The Legend of Zelda: Link's Awakening DX.

Game Boy Advance, 2001

The Game Boy Advance (see Figure 5.3) is significantly more powerful than any previous Game Boy model, with nearly twice the screen resolution, 10 times more memory than the Game Boy Color, and a speedy ARM7-based RISC CPU (more than twice as fast as the Game Boy Color). In addition, the Game Boy Advance incorporates the original Z80 CPU from the Game Boy Color, providing for *complete* backward compatibility with all previous Game Boy cartridges. Basically, the Game Boy Advance detects the type of cartridge that has been inserted and boots up on either the ARM7 or the Z80 CPU. If you think about it, how many other video game consoles today are capable of running games from 1989—original games, in their original cartridges? None! Only the Game Boy Advance (and the Game Boy Color before it) is capable of this feat.

The hacks in this chapter focus on this original GBA model.

Figure 5.3 Nintendo Game Boy Advance, 2001

Game Boy Advance SP, 2003

The Game Boy Advance SP (see Figure 5.4) is a variation of the Game Boy Advance with an internally lighted screen, long-lasting rechargeable battery (built in), and folding clamshell design. In all other respects, the SP has the same internal components as the Game Boy Advance, and the changes are merely cosmetic. The rechargeable battery is internal (and therefore not replaceable without disassembly), but it does promise longer life than the AA batteries used in a Game Boy Advance. One interesting aspect of the SP is that, when the screen is completely opened, it closely resembles the shape of the original Game Boy!

Figure 5.4 Nintendo Game Boy Advance SP, 2003

A Very Brief History of Nintendo

Nintendo did not always make video games. In fact, the name *Nintendo* was established in 1951, and the company itself was actually founded in 1889. That company was called Marufuku Company, started in Japan by Fusajiro Yamauchi. Marufuku manufactured stylish Japanese playing cards (known as Hanafuda), and in 1902 Yamauchi expanded into other types of playing cards.

In 1951, Marufuku Company was renamed Nintendo Karuta (which translates to *Nintendo Playing Cards*). The word *Nintendo* consists of three kanji characters that translate to *leave luck to heaven*. Nintendo got into the toy manufacturing business in the 1970s, building toys such as light guns. Nintendo's first videogame machine was actually a license to sell the Magnavox Odyssey in Japan, in 1975. Nintendo's experience with the Odyssey helped the company develop its own videogames. In 1977, in a joint venture with Mitsubishi, Nintendo created the Color TV Game 6 and Color TV Game 15 systems, which were clones of the popular Pong videogame.

In 1978, Nintendo built its first arcade game, a small table-sized cabinet called Computer Othello, which featured 10 buttons and was overly complex for the general game-playing public. This game was not a serious competitor for Bally Midway's Space Invaders, which also came out in 1978 and was hugely successful, but it did help launch Nintendo's arcade game division. At the time, hundreds of thousands of Space Invaders machines were produced, and the game's popularity led to a shortage of the Japanese 100-Yen coin used to play the game. In 1980, Nintendo's Radar Scope was introduced; the game sold poorly, leaving Nintendo with 2,000 unsold cabinets.

Hiroshi Yamauchi, then president of Nintendo, decided to replace Radar Scope boards with a new game rather than recall all the unsold cabinets. Enter Shigeru Miyamoto. Yamauchi hired Miyamoto as a staff artist as a personal favor for a friend, Miyamoto's father. Yamauchi gave Miyamoto the job of writing a game for the Radar Scope boards so that a simple upgrade could be performed to the existing cabinets.

Miyamoto had no idea how to write a game, so he worked with the video game designers to translate his designs into a game. His design called for a small, animated carpenter that could run, jump, climb ladders, and defeat a gorilla named Donkey Kong to save a blonde girl (who was held captive by Donkey Kong at the top of the screen). Donkey Kong was hugely popular in the United States and firmly established Nintendo as a force to be reckoned with in the arcade business. Donkey Kong also set the stage for Mario, arguably the greatest and most well-known video game character in history. Although originally named Jumpman and originally a carpenter, the likeness of Mario was first established in the seminal Donkey Kong video game. Jumpman the carpenter was transformed into Mario later in Donkey Kong Jr. and then gained a new profession as a plumber in Mario Bros.

In 1983, Nintendo introduced the 8-bit NES in Japan, with launch titles Donkey Kong, Donkey Kong Jr., and Popeye. Timing was very important because the United States had just gone through a video game crash, largely due to the failure of Atari and other game console vendors (primarily because of poor management decisions that miscalculated demand). Nintendo didn't release an American version of the Famicom, dubbed the Nintendo Entertainment System (NES), until 1985. When Nintendo's first console did finally reach U.S. shores, Nintendo had 90 percent of the Japanese market. The NES quickly dominated the U.S. market as well. The success of the NES established Nintendo as the firm leader in the console business, and the company poured its resources into this

division, dropping out of the arcade business altogether. (For more information on this console, see Chapter 7: Nintendo NES.)

Nineteen eighty-nine saw the introduction of the first Game Boy, which would become the beginning of Nintendo's domination of the handheld market. Gunpei Yokoi, who had designed the popular Game & Watch handheld games for Nintendo earlier in the 1980s, designed the Game Boy console.

For a more complete history of Nintendo, take a look at *Game Over: Press Start to Continue*, by David Sheff (Cyberactive Media Group, 1999, ISBN 0-9669617-0-6).

Opening the GBA Console

Two hacks in this chapter require you to open the GBA console to gain access to the internal components and circuitry. So, that's what we'll do in this hack. Opening the system is simple, but keeping track of where all the screws go can be confusing.



Preparing for the Hack

To complete this hack, you will need the following tools:

- Jeweler's size Phillips screwdriver
- NES Tri-Wing security screwdriver

The Tri-Wing security screwdriver fits onto specially shaped screws (see close-up shown in Figure 5.5) used to keep the Game Boy Advance, Game Boy Color, and Game Boy Pocket systems together. (These screws are also used in some of Nintendo's accessories, such as the Game Boy e-Reader.) The tool is available for around \$5 on eBay (do a search for Game Boy Tri Wing), GameConsoleRepair.com, and many other online video game retail stores.

Figure 5.5 Close-Up of the NES Tri-Wing Security Screw



Performing the Hack

Perform the following:

WARNING: HARDWARE HARM



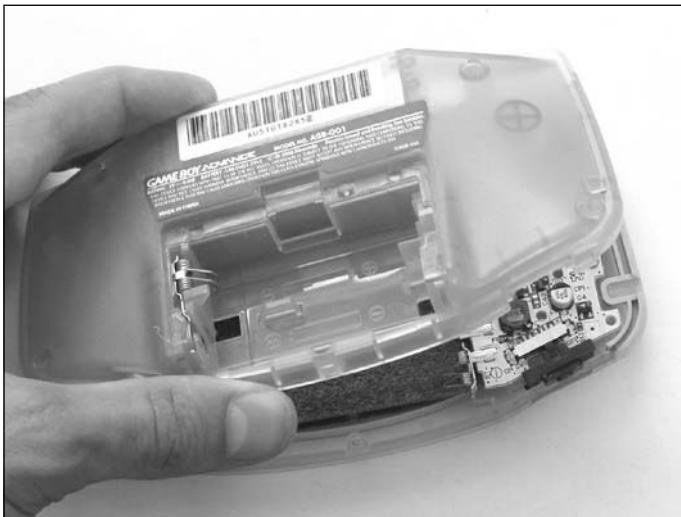
The GBA circuit board contains many electrostatic-sensitive components. You must always be grounded before touching the inside of your GBA or you risk damaging these components. The easiest way to ground yourself is to buy an antistatic wrist strap and connect it to a grounded source. If you do not have a wrist strap, you can ground yourself by touching a piece of metal (such as the edge of your desk, if it is metal) before touching the GBA's circuitry.

1. To open the GBA handheld, which will give you access to all of the internal circuitry, you first need to flip the case upside down, open the battery cover, and remove the two AA batteries (if installed). If you have a game cartridge inserted into the system, remove that, too.
2. Remove the seven screws holding the case together (see Figure 5.6). Of the seven screws, six require the special Tri-Wing security screwdriver, and one requires the standard jeweler's size Phillips screwdriver.

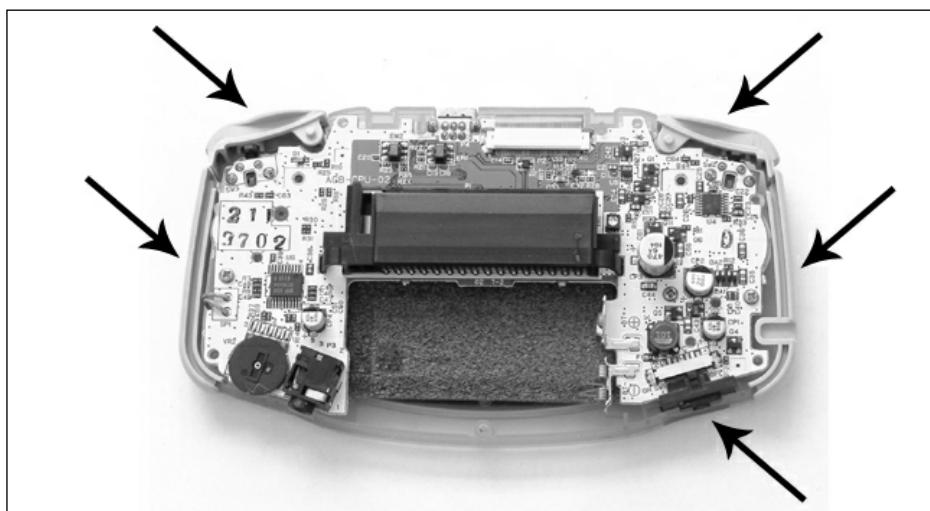
Figure 5.6 Backside of the GBA Showing Screw Locations



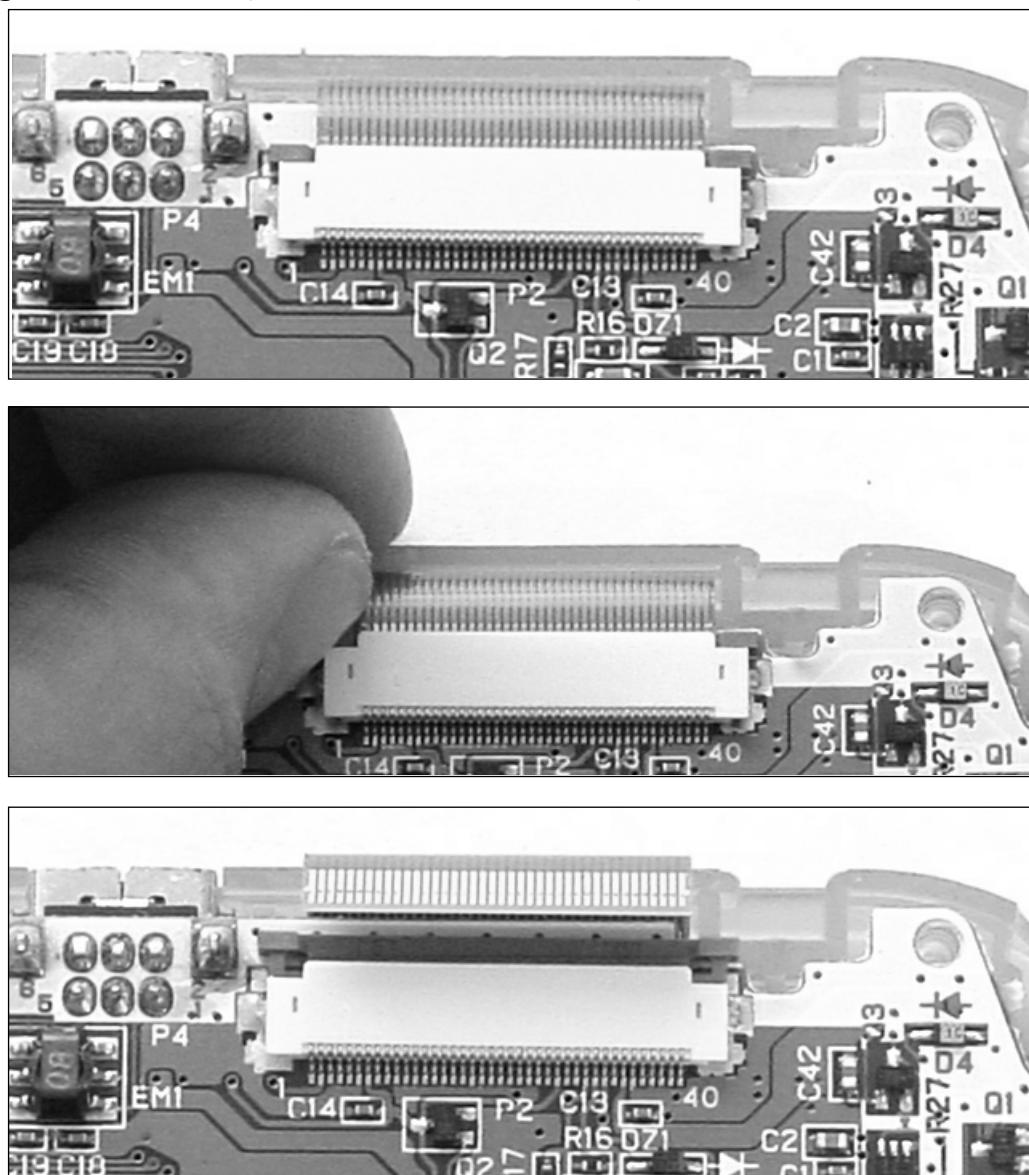
3. The case consists of two plastic pieces, the top and the bottom. After you remove the screws, carefully lift the bottom of the GBA housing straight upward, separating it from the top (see Figure 5.7).

Figure 5.7 Separate the Bottom and Top Halves of the Console

- With the bottom half of the case removed, take off the plastic left and right finger triggers, power switch cover, and left and right bumpers (see Figure 5.8) and set them aside.

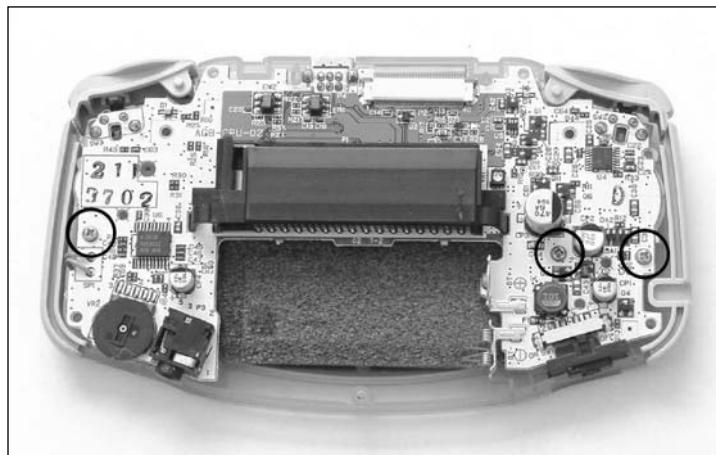
Figure 5.8 GBA Bottom Circuitry with Finger Triggers, Power Switch Cover, and Bumpers Denoted

- Now carefully remove the LCD flex cable from its connector on the circuit board (see Figure 5.9). To do this, use your fingernail to push the small brown plastic piece away from you. This plastic piece inside the connector holds the flex cable in place. When the plastic piece is “unlocked,” you will easily be able to pull the flex cable out of the connector.

Figure 5.9 Removing the LCD Cable: Before, During, and After

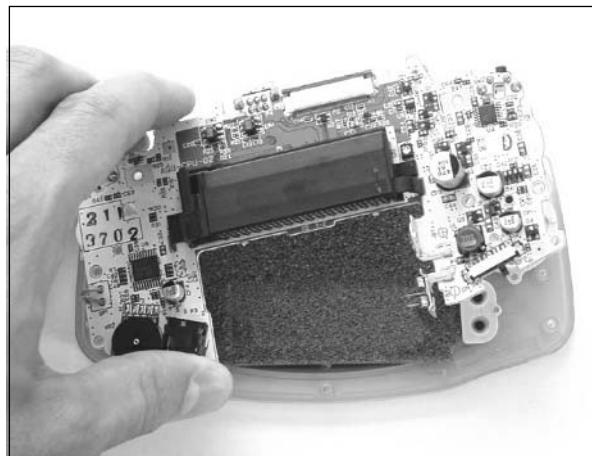
6. Next, we have to remove the front housing, which will give us access to panel buttons and the front of the circuit board. Do so by removing the three screws holding the circuit board to the front housing, denoted in Figure 5.10.

Figure 5.10 Remove the Three Screws from the Circuit Board

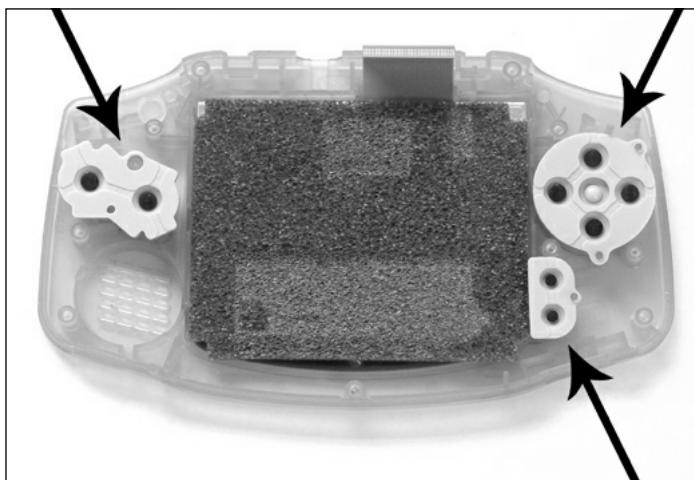


7. With the screws removed, gently lift the circuitry away from the bottom housing (see Figure 5.11).

Figure 5.11 Remove the Circuit Board from the Bottom Housing



8. From the top housing, if necessary, you can remove the rubber switch membranes for the four-direction control pad and the Start, Select, A, and B buttons (see Figure 5.12). You can also remove the plastic pieces in the A and B button slots.

Figure 5.12 Remove the Rubber Switch Membranes and Plastic Pieces (Optional)

- With the circuit board removed, you should be left with two housing pieces (one still containing the LCD—see the “Light Up Your LCD with the GBA Afterburner Mod” hack later in this chapter for information on how to remove the LCD), various plastic and rubber pieces, and the complete circuit board (see Figures 5.13, 5.14, and 5.15).

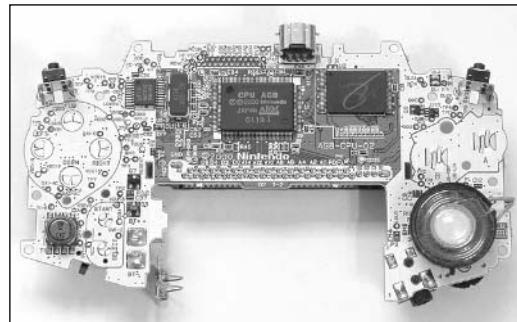
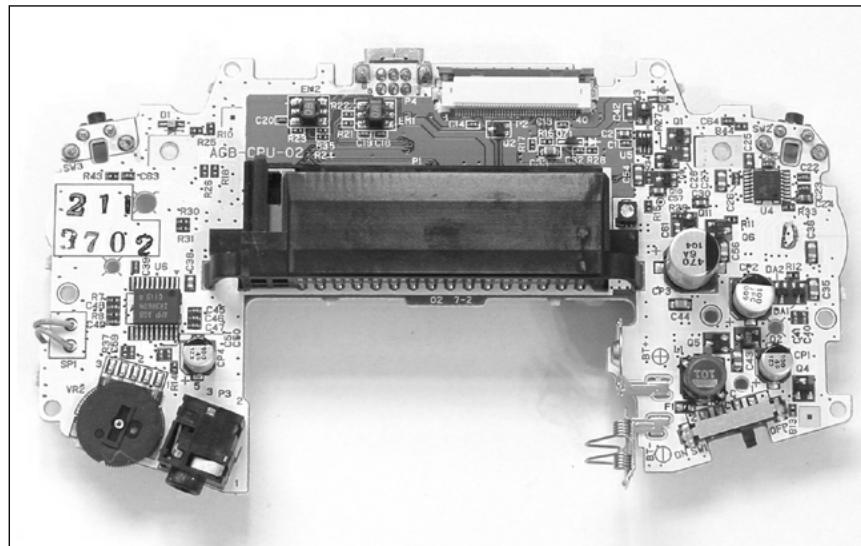
Figure 5.13 Completely Disassembled GBA**Figure 5.14 GBA Circuit Board, Front**

Figure 5.15 GBA Circuit Board, Back

To put the console back together, simply follow the disassembly steps in reverse order:

1. First, if applicable, insert the plastic A and B buttons back into their respective slots (when you're looking at the front of the GBA console, the A button should be on the right and the B button should be on the left), and reattach all the rubber switch membranes for the buttons and four-direction control pad.
2. Place the circuit board into the front housing, taking care not to crush the LCD flex cable.
3. Screw the three screws back into the circuit board.
4. Insert the LCD flex cable into the connector and press the brown plastic piece inward to secure the cable.
5. Replace the power switch cover, left and right finger triggers, and left and right bumpers.
6. Attach the bottom half of the GBA housing and secure it into place with the seven screws. The only Phillips head screw goes into the hole inside the battery compartment.

Your GBA should now be reassembled!

Replacing the Display Lens

Due to the portable nature of the GBA, it is only a matter of time before your screen will become scratched or dirty enough to affect game play and the enjoyment of your console. This simple hack will guide you through the process of replacing the plastic screen cover of the GBA that is used to protect the LCD display. If done correctly, replacing the screen cover will help make your display appear as it did when you first bought your GBA.

This hack is extremely simple and serves as a good starting point for aspiring hardware hackers. There isn't much to be done—you don't even have to open your GBA—but the end result is very satisfying.



Preparing for the Hack

The only part required for this hack is a replacement GBA display lens (sometimes referred to as a *screen cover*; see Figure 5.16). You can obtain a brand-new, original Nintendo display lens from Goldberg-Mods (www.goldberg-mods.co.uk) or a new, third-party display lens on eBay (do a search for Game Boy Lens), GameConsoleRepair.com, and many other online video game retail stores.

The only tool you'll need is an X-ACTO knife or hobby blade to help remove the original display lens.

Figure 5.16 GBA Replacement Display Lens



Performing the Hack

Perform the following:

WARNING: HARDWARE HARM



When you perform the GBA display lens replacement hack, be sure that your workspace is clean and completely free of dust and dirt. Any small specs of debris that come close to or in contact with the new screen or the LCD panel could cause scratching and undesired display artifacts.

1. We'll be working with the front half of the console housing that contains the plastic protective screen cover (see Figure 5.17). You don't need to open your GBA, although we did for the pictures in this section.

Figure 5.17 GBA Top Housing with Display Lens



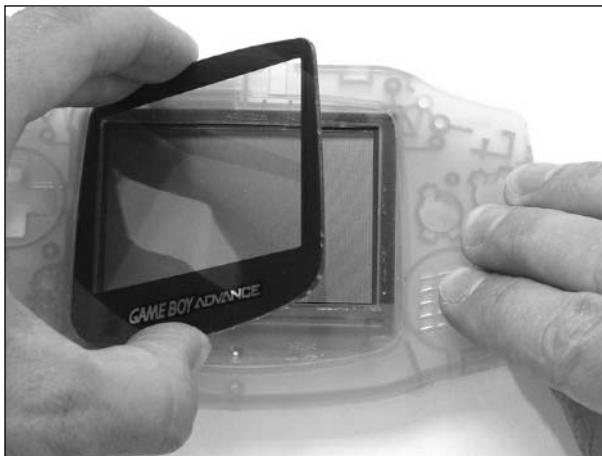
2. The display lens is attached to the front of the GBA with a very strong adhesive tape. Using the X-ACTO knife, carefully loosen the adhesive by gently prying along the edge of the lens (between the lens and the GBA's plastic housing; see Figure 5.18). You'll probably want to start at a corner and move slowly along one of the edges. Be careful to not dent or otherwise disfigure the GBA's housing. As the display lens loosens from the housing, make sure your X-ACTO knife does not slip underneath and touch the LCD screen below!

Figure 5.18 Prying the Original Display Lens Away from the Housing



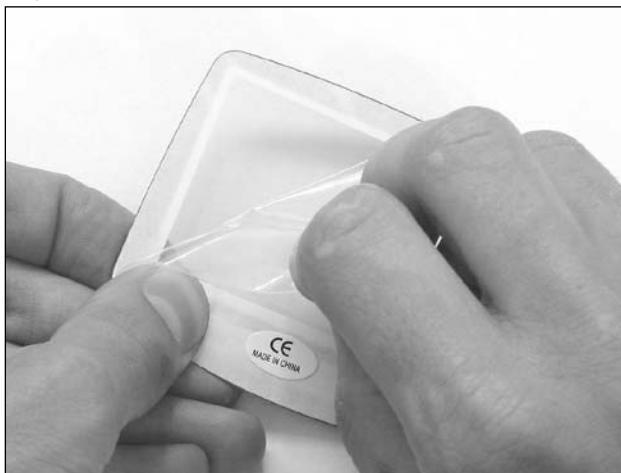
3. Completely remove the display lens from the GBA (see Figure 5.19) and discard it.

Figure 5.19 Removing the Display Lens



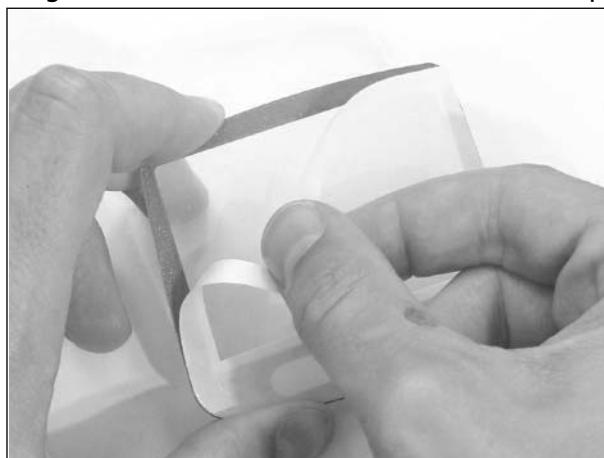
4. Remove the protective film from the back side of the new GBA display lens (see Figure 5.20). Once the protective film is removed from the new screen cover, be very careful not to scratch the screen or get any dirt, dust, or fingerprints on it, since they would affect the visual quality of the GBA once you reassemble the material.

Figure 5.20 Peeling the Protective Film from the Back of the New Display Lens



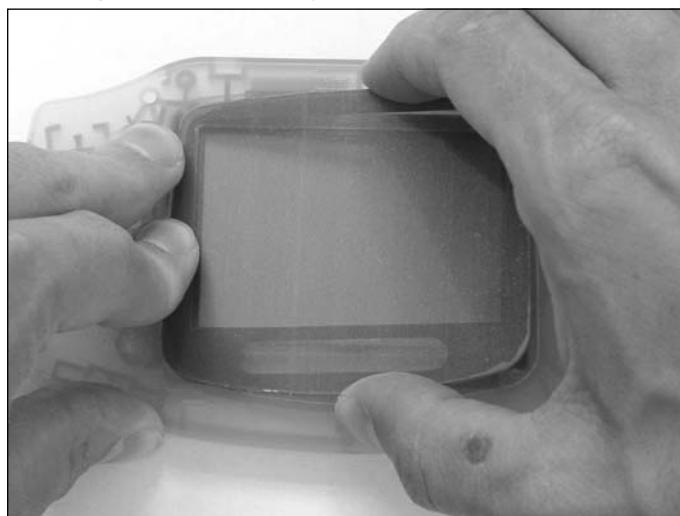
5. The backside of the display lens also has a protective film over the adhesive. Simply peel this protective film off to expose the adhesive (see Figure 5.21). Be careful not to get any fingerprints on the actual clear lens.

Figure 5.21 Exposing the Adhesive on the Back of the New Display Lens

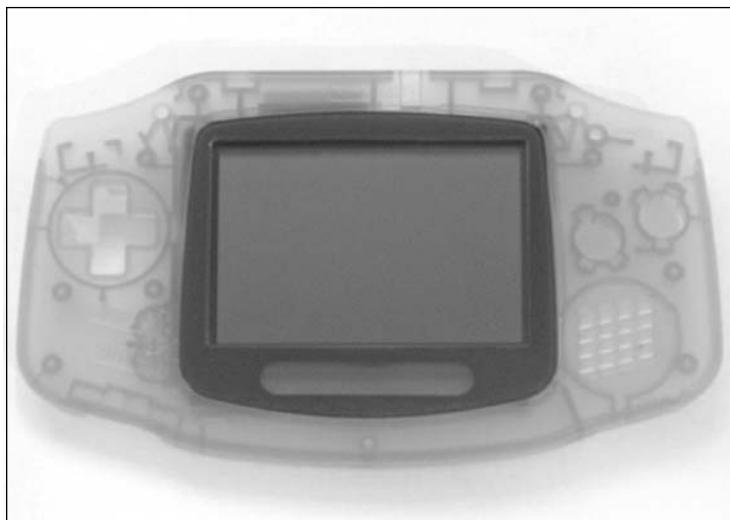


6. Now place the new lens into the location of the old one. Secure it to the housing by pressing firmly around all four edges, making sure to push in the four corners as well (see Figure 5.22).

Figure 5.22 Attaching the New Display Lens onto the GBA



7. Finally, remove the protective film from the front side of your new GBA display lens and enjoy your new, crystal clear screen (see Figure 5.23)!

Figure 5.23 New Display Lens

Light Up Your LCD with the GBA Afterburner Mod

One disadvantage to the Game Boy Advance is the lack of any built-in lighting solution for the LCD. (The Game Boy Advance SP is the first in the Game Boy family to have a backlight.) The GBA screen is difficult to see in low lighting conditions, such as indoors or at night. This lack of lighting can make for sore eyes and diminished game-play value.

This hack guides you through the process of installing the extremely popular Triton Labs' GBA Afterburner internal lighting kit. This essential add-on consists of an internal flat-surface LED light guide that is positioned in front of the LCD, providing a remarkable improvement in screen visibility, regardless of the lighting conditions. When you're done, you will be left with a high-quality, front-lit LCD display with optional adjustable brightness control.



Preparing for the Hack

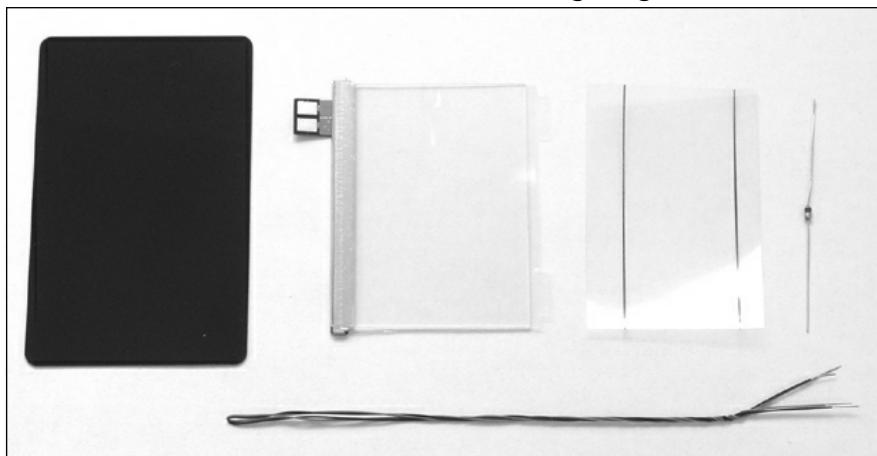
To complete this hack, you will need the following parts:

- **Afterburner GBA Internal Lighting Kit** In January 2004, Triton Labs, the manufacturer of the Afterburner, ceased production on the kit. Your best bet for an Afterburner is from eBay (do a search for Game Boy Afterburner). You can also try your luck with the dozens of online video game retail stores to see if they have any remaining stock of the kits. The kit was relatively inexpensive (under \$30) when it was in production; the price has increased slightly since then (to approximately \$50 or \$60). Don't give up searching for an Afterburner kit—your hard work will pay off because the hack will work wonders on your GBA screen!

The contents of the kit (see Figure 5.24) include:

- A plastic card
- Afterburner front light module (leave the protective liners in place until directed to remove them)
- Antireflective film (leave the protective liners in place until directed to remove them)
- A 44 ohm, 1/8 watt resistor
- One-foot lengths of black and red wire

Figure 5.24 Contents of the Afterburner Internal Lighting Kit



- **Afterburner GBA Case Housing (optional)** You can obtain a brand-new replacement front housing on eBay (do a search for Afterburner Case), GameConsoleRepair.com, and many other online video game retail stores. This case (see Figure 5.25) has a professionally installed potentiometer for the optional adjustable brightness control for the Afterburner hack. We recommend that you use one of these case housings for the hack, since you will save time by not needing to modify the original GBA case to fit the Afterburner. (These new cases were designed specifically to fit the module.) You have a number of different colors to choose from, so you can match the color to your original GBA color if you like, or customize it with a different color: platinum, gold, glacier, indigo, fuchsia, white, black, or orange.

Figure 5.25 The Afterburner GBA Case Housing (Optional)

You'll also need the following tools:

- Jeweler's size Phillips screwdriver (to open the GBA case)
- NES Tri-Wing security screwdriver (to open the GBA case)
- Soldering iron and solder
- Wire strippers
- X-ACTO knife, hobby blade, or Dremel tool (to modify some of the plastic within the GBA housing); these tools are not necessary if you are using the Afterburner GBA Case Housing

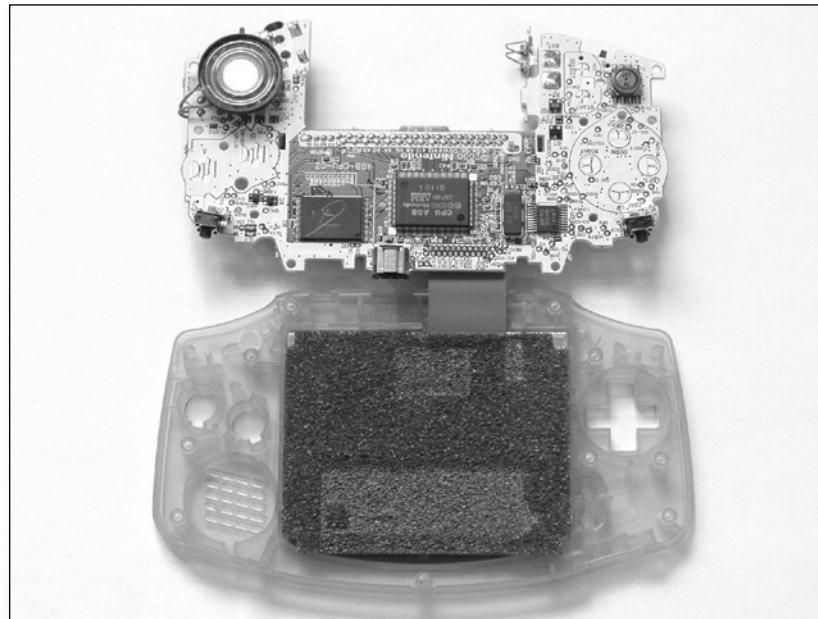
We will use the optional Afterburner GBA Case Housing for this hack. If you choose to keep the original GBA housing, you will need to make modifications (detailed in the following process) to the plastic housing, to allow the Afterburner module to fit properly.

Performing the Hack

Perform the following:

1. Open your GBA as described in the “Opening the GBA Console” section earlier in this chapter. Skip Step 5 of that process, since we will be leaving the flexible LCD cable connected to the GBA circuit board for this hack. Your open unit, ready for the hack, should resemble the one shown in Figure 5.26.

Figure 5.26 GBA Opened and Ready for Modification

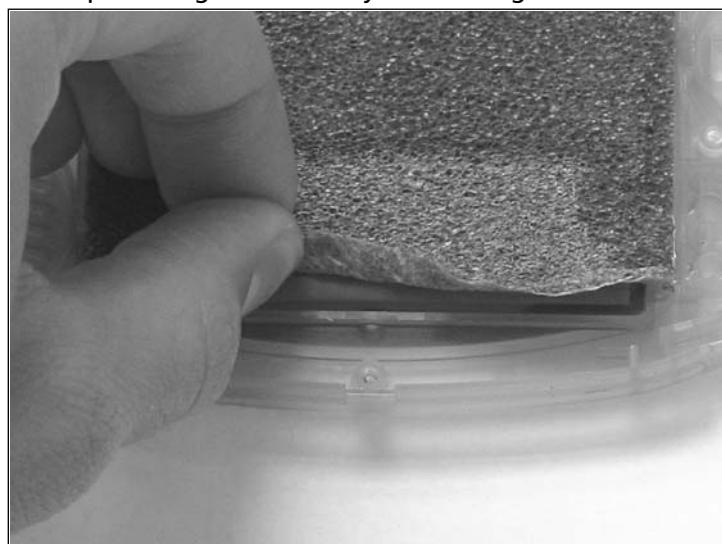


Removing the LCD

To remove the LCD, perform the following:

1. Peel up the bottom edge of the grey cushioning (located on top of the LCD; see Figure 5.27).

Figure 5.27 Peel Up the Edge of the Grey Cushioning



2. Carefully remove the LCD from the GBA front housing using the blank plastic card. The LCD is attached to the front of the GBA with a strong adhesive cushion. Slide the plastic card underneath the edge of the LCD and gently pry it up (see Figure 5.28). As you move forward toward the front of the unit, the adhesive holding the LCD to the housing will loosen, and you will be able to remove the display (see Figure 5.29).

Figure 5.28 Use the Plastic Card to Loosen the LCD from the Housing

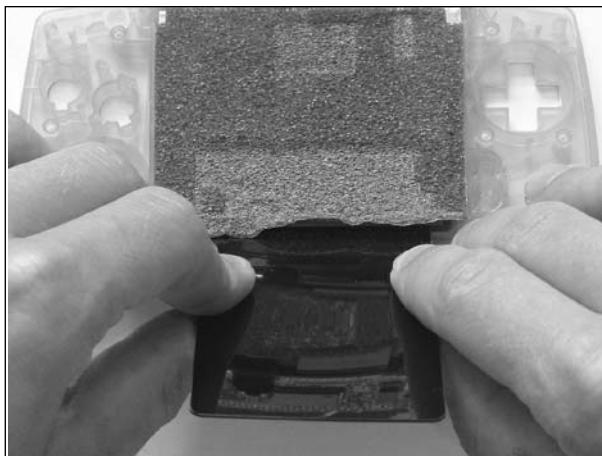
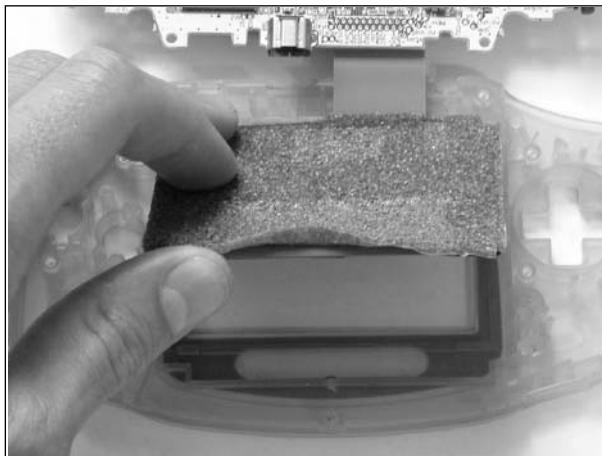
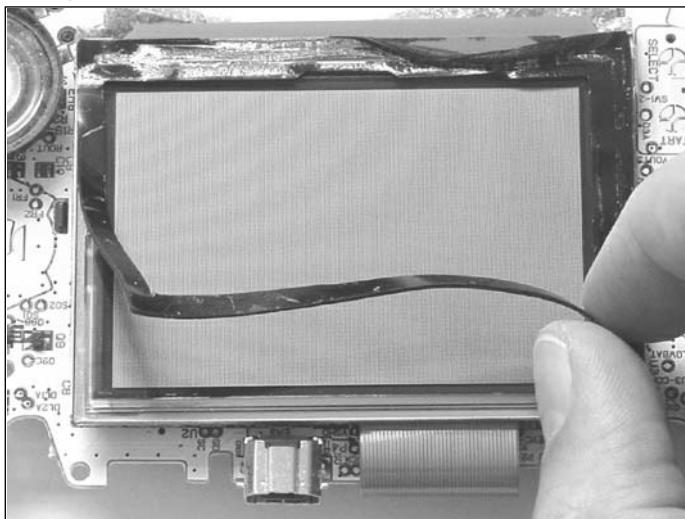


Figure 5.29 Removing the LCD



3. Peel off the adhesive cushion from the face of the LCD (see Figure 5.30) and place it aside for later in the hack. Take care to not damage or stress the LCD during this step.

Figure 5.30 Peeling Off the LCD's Adhesive Cushion

Preparing the GBA Housing

For the Afterburner module to fit properly, you need to make some simple modifications to the housing of the original GBA. If you are using the optional Afterburner GBA Case Housing, you can skip this section and go directly to the following section, “Preparing the LCD.”

Perform the following:

1. Remove the plastic display lens from the front of the housing. The display lens is attached to the front of the GBA with a very strong adhesive tape. Using the X-ACTO knife, carefully loosen the adhesive by gently prying along the edge of the lens (between the lens and the GBA’s plastic housing; refer back to Figure 5.19). This step is detailed in the earlier “Replacing the Display Lens” hack.

With the lens removed, the front housing will be ready for modification (see Figure 5.31).

Figure 5.31 Original GBA Housing Ready for Modification

2. Two sections of plastic must be modified. First, use the X-ACTO knife or Dremel tool to cut a 15mm wide notch out of the bottom plastic rail (denoted in Figure 5.32). The width of the notch is not critical, provided that the “tail” of the Afterburner light (the circuit board tab containing the two solder pads) can fit through it. The notch should *not* go entirely through the housing to the other side—it should only be cut flush with the inside of the housing (see Figure 5.33).

Figure 5.32 Cut a 15mm Notch from the Original GBA Housing

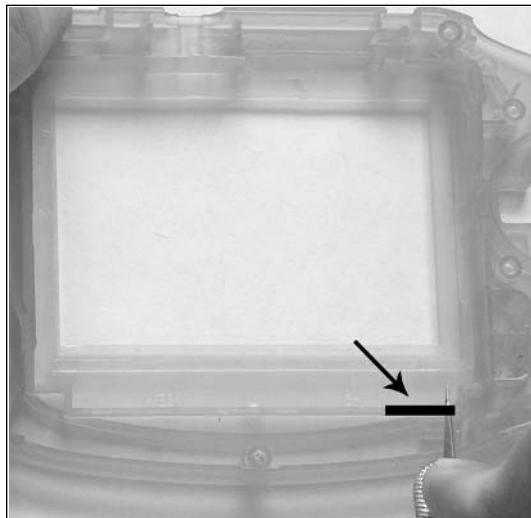
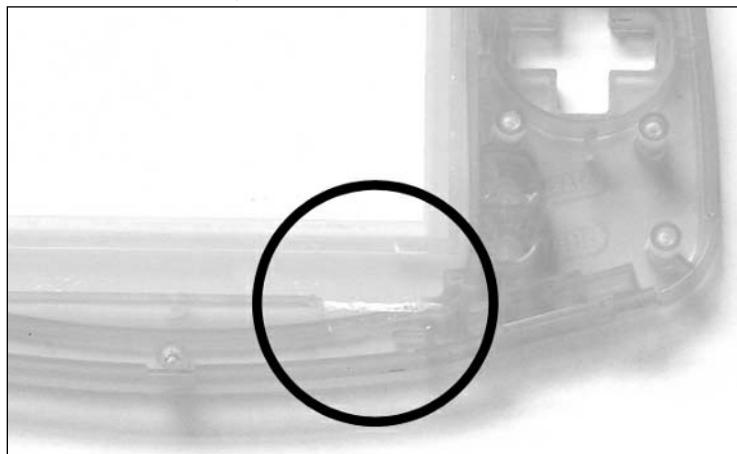


Figure 5.33 Notch Successfully Cut



3. Use the X-ACTO knife or Dremel tool to completely remove a section of the housing (denoted in Figure 5.34). The cut should be approximately 6.5mm wide and 67mm long,

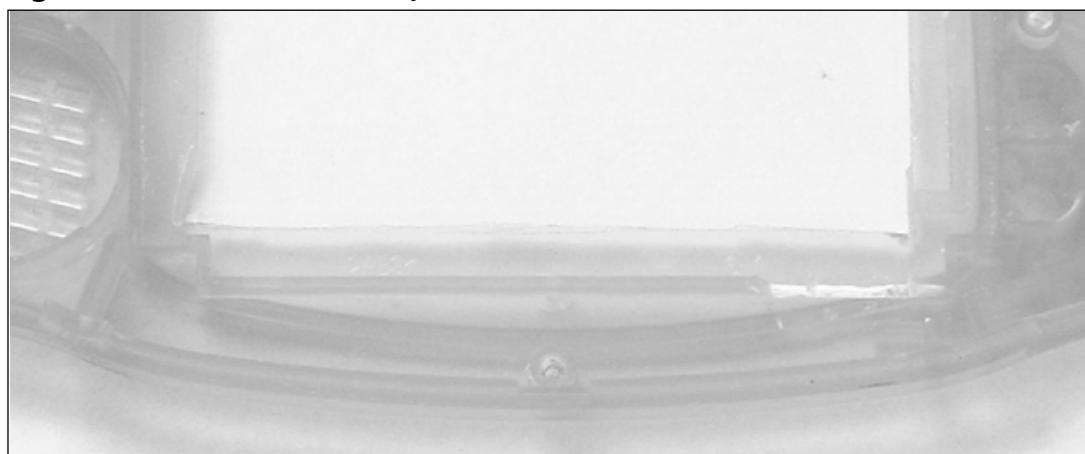
located on the bottom edge of the LCD area. This cut *should* go entirely through the housing to the other side (see Figure 5.35). You might want to smooth out the area with the X-ACTO knife or some sandpaper if any of the plastic on the housing is jagged from the cutting.

Before continuing with the hack, ensure that the Afterburner module fits snugly into the area where the LCD was previously located. If it does not, trim away any plastic that is still in the way.

Figure 5.34 Remove a Length of Plastic from the Bottom Edge of the LCD Area



Figure 5.35 Plastic Successfully Removed

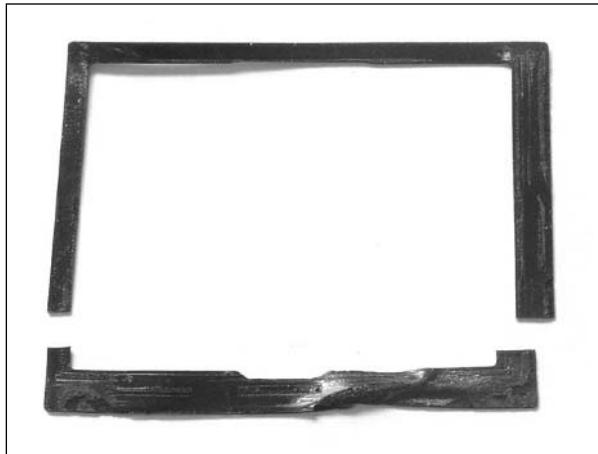


Preparing the LCD

The next section in this hack will prepare the GBA's LCD and housing. Perform the following:

1. Using the X-ACTO knife, cut the adhesive cushion as shown in Figure 5.36. Discard the bottom piece.

Figure 5.36 Cut the Adhesive Cushion



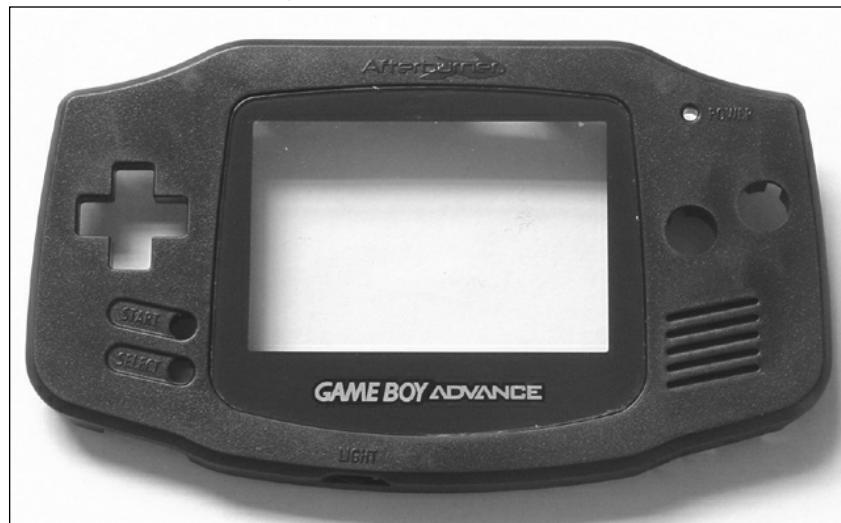
2. Now replace the top piece of the adhesive cushion into the GBA front housing (see Figure 5.37). If you are using the optional Afterburner GBA Case, as we are demonstrating in this hack, you can discard the original GBA housing and use the new one from this point forward.

Figure 5.37 Replace the Top Piece of Adhesive Cushion into the GBA



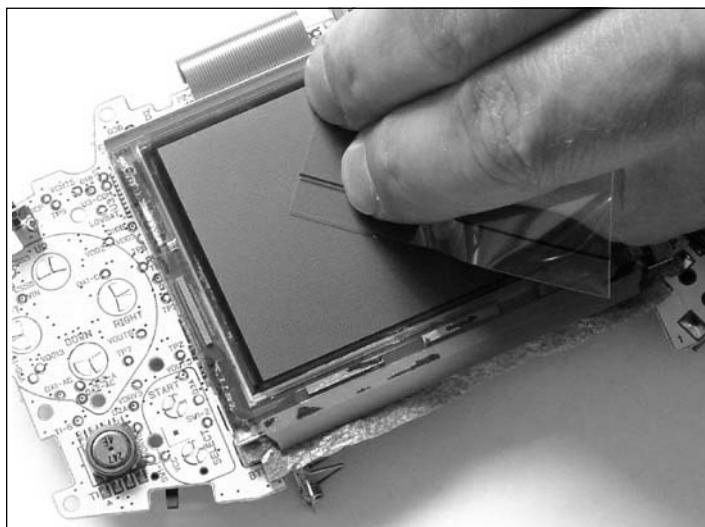
3. Place the plastic display lens from the front of the original GBA housing (or a new replacement display lens) onto the new Afterburner GBA Case (see Figure 5.38). The display lens attaches to the front of the GBA with a very strong adhesive tape. If this adhesive is weak from when you removed the lens in Step 5, you can use some double-sided tape to help keep the lens in place.

Figure 5.38 Attach the Display Lens



4. Next we need to attach the antireflective (AR) film to the GBA's LCD. First, determine which side of the AR film is “sticky” without touching that side of the film or otherwise contaminating it in the process. Both sides of the AR film are protected with liners that shouldn't be fully removed at this step.

To determine the sticky side, carefully peel back one corner of the liner and press it against the LCD surface (see Figure 5.39). If it sticks, that's the sticky side. If it doesn't stick, the other side of the AR film is the sticky side. Make a mental note of which side is sticky for the next step.

Figure 5.39 Finding the “Sticky” Side of the Antireflective Film**WARNING: HARDWARE HARM**

Before performing the next step, be sure to read it completely. If it is followed incorrectly, you could end up with air bubbles on your LCD screen that will affect the display quality when the hack is done. Ensure that the LCD surface is completely free of dust and fingerprints before continuing.

5. Laminate the AR film onto the LCD screen. The sticky side, as you determined in the previous step, should be placed face down against the LCD screen. Begin the lamination process by peeling off a small portion of the liner and affixing just the exposed edge of the film face down against the LCD screen. Then gradually release the liner from the AR film while following behind with the blank plastic card (see Figure 5.40). Use enough pressure on the plastic card to ensure that the film sticks to the LCD and that there are no residual air bubbles. Take care to not touch the sticky side of the AR film with your fingers!

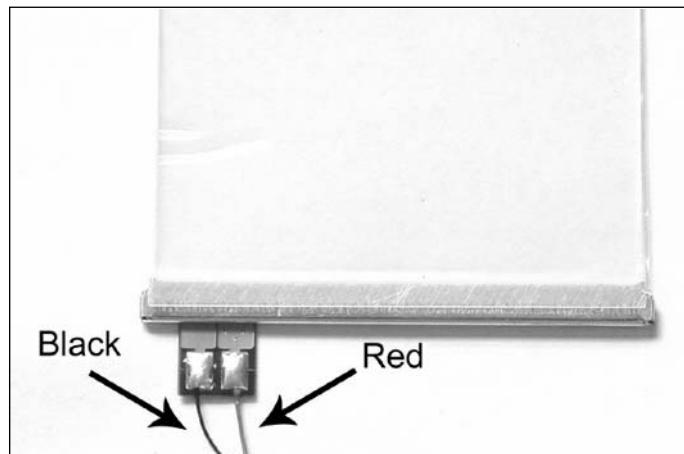
Figure 5.40 Laminating the Antireflective Film to the LCD

6. After the AR film has been successfully affixed to the LCD, remove the protective liner from the top of the AR film.

Preparing the Afterburner Module

The next section in this hack will prepare the Afterburner module for placement into the GBA. Perform the following:

1. Solder the red and black wires to the leads at the end of the Afterburner module. The black wire should be soldered to the left pad, and the red wire should be soldered to the right pad (see Figure 5.41). It could help to “tin” the pads of the Afterburner module by applying solder directly to them before soldering on the wire. This will give you a stronger solder joint.

Figure 5.41 Solder Wires to the Afterburner Module

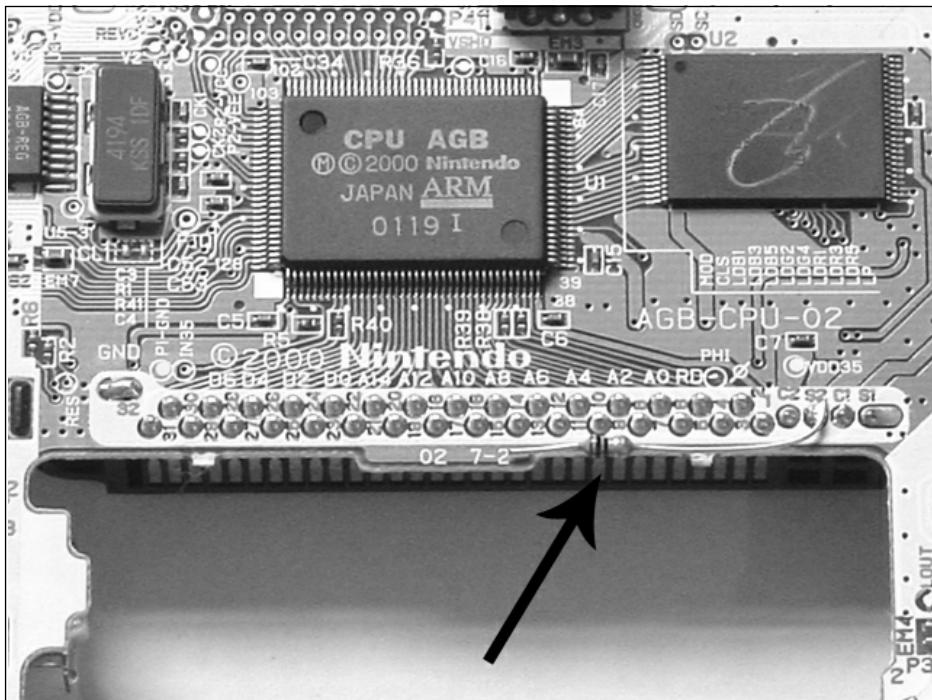
2. Next, bend and clip the leads of the resistor included with the Afterburner kit, as shown in Figure 5.42. It doesn't matter which side is which—the polarity of the resistor does not matter.

Figure 5.42 Bend and Clip the Resistor



3. Solder the longer, bent end of the resistor to the pad labeled S2 on the front of the GBA circuit board (see Figure 5.43). Position the resistor in the crevice of the circuit board to prevent internal stress to the LCD when the console is reassembled.

Figure 5.43 Solder One Side of the Resistor to the S2 Pad on the GBA

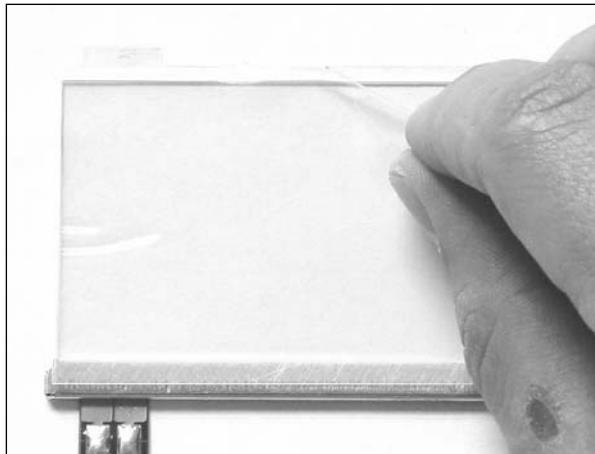


Installing the Afterburner Module

Now it's finally time to install the Afterburner module into your GBA. Perform the following:

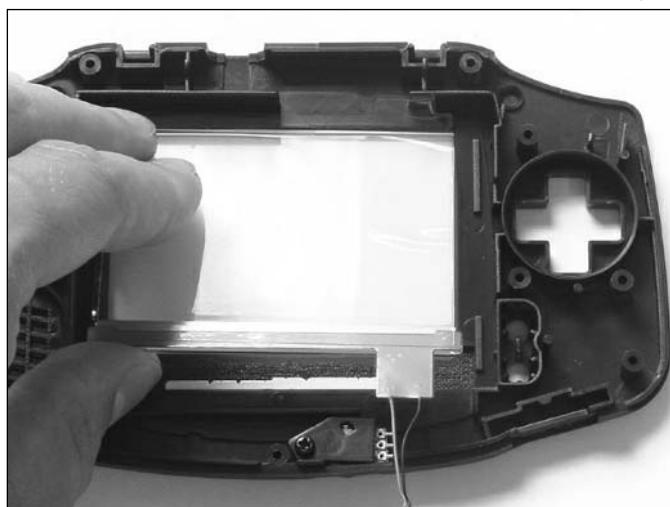
1. Ensure that the surfaces of the plastic display lens and the LCD screen (with the affixed AR film) are free of dust and fingerprints. Then remove the protective liner from the front surface of the Afterburner module (see Figure 5.44).

Figure 5.44 Peeling the Protective Film from the Front of the Afterburner



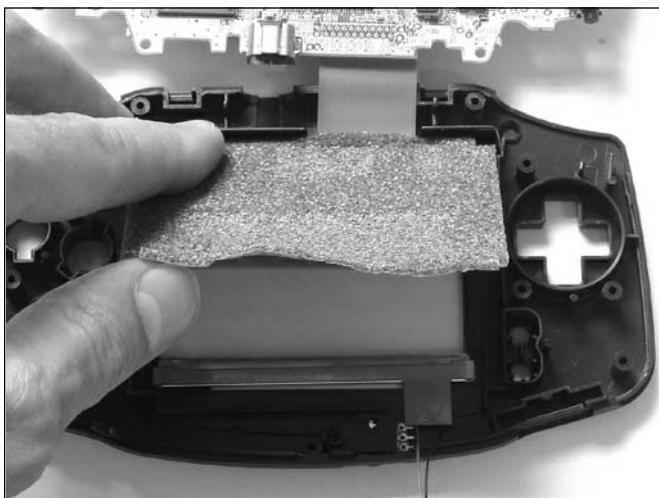
2. Next, without touching the Afterburner's exposed front face, place it into the opening of the GBA housing (or the Afterburner GBA Case Housing; see Figure 5.45). Push firmly to secure the module into place.

Figure 5.45 Insert the Afterburner Module into the GBA Housing



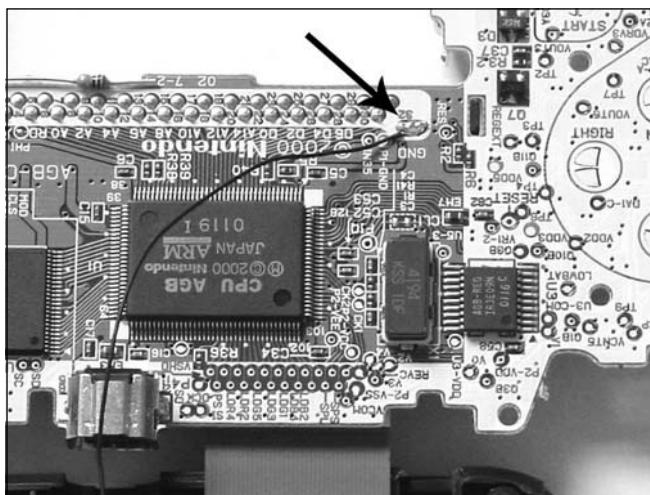
3. With the Afterburner successfully in place, you can remove the protective liner from the back surface of the module.
4. Now replace the LCD module back into the housing, sitting on top of the Afterburner module (see Figure 5.46). Again, take care to not touch the exposed surfaces of the LCD screen or Afterburner.

Figure 5.46 Replace the LCD into the GBA Housing



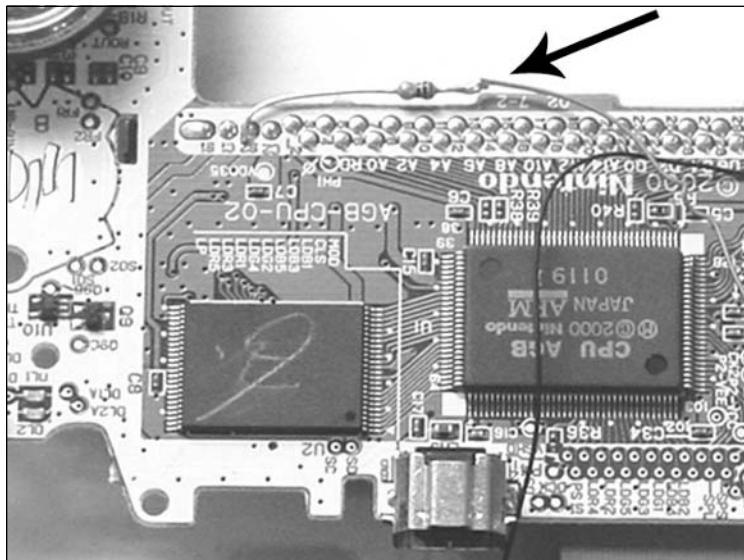
5. Solder the black wire coming from the Afterburner module to the pad labeled GND on the front of the GBA circuit board (see Figure 5.47). You may choose to clip and restrip the black wire to make it shorter before soldering it into place.

Figure 5.47 Solder the Black Wire to the GND Pad on the GBA



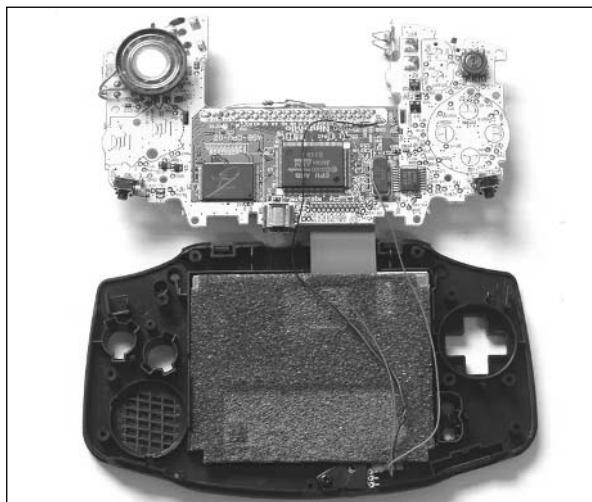
6. Solder the red wire coming from the Afterburner module to the unconnected end of the resistor (see Figure 5.48). You may choose to clip and restrip the red wire to make it shorter before soldering it into place.

Figure 5.48 Solder the Red Wire to the Resistor



7. If you will be installing the brightness control, skip directly to the following section, “Adding the Brightness Control.” Otherwise, you are done with the hack! Your completed Afterburner installation (before you close up the case) should resemble the one in Figure 5.49.

Figure 5.49 Completed Afterburner Installation (No Brightness Control)



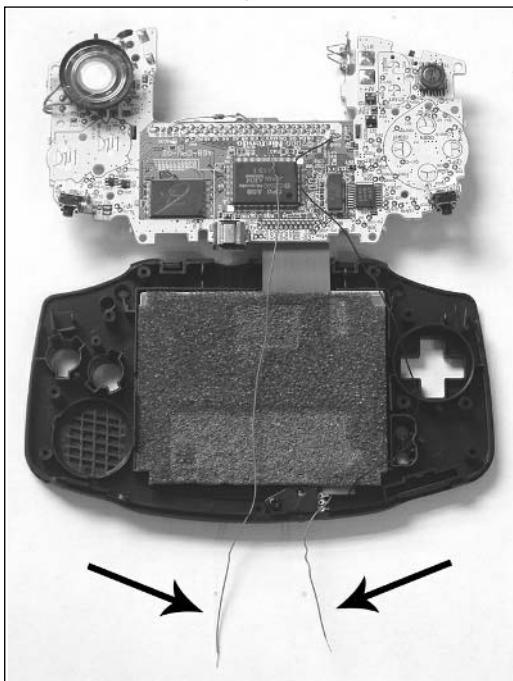
Adding the Brightness Control (Optional)

This section assumes that you are using the Afterburner GBA Case Housing, which contains the brightness dial (a potentiometer) built into the housing. If you are using the original GBA case, you will need to mount the potentiometer onto the case and drill three holes to allow the pins to stick through to the inside of the case.

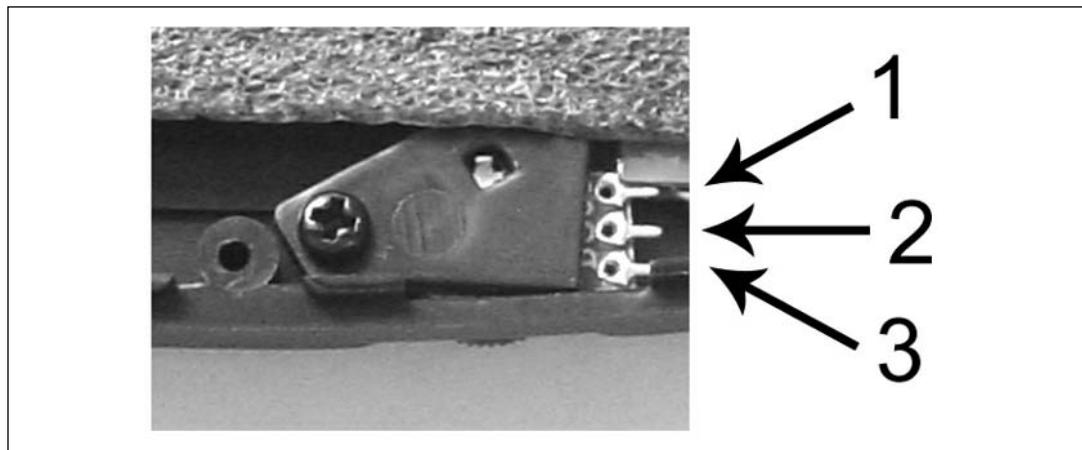
Perform the following:

1. Cut the red wire into two halves and strip each end as necessary (see Figure 5.50). The length of wire that is soldered to the resistor should be longer, since it needs to reach down to the potentiometer mounted below the Afterburner module at the bottom of the case. The length of wire connected to the Afterburner should be shorter, since it only needs to connect to the potentiometer directly below it.

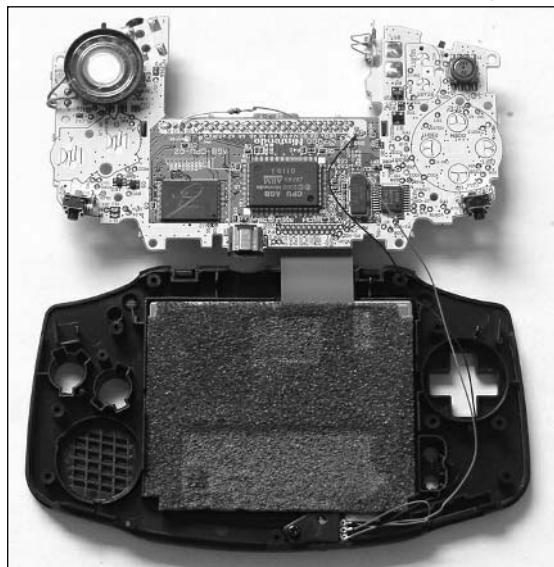
Figure 5.50 Red Wire Cut into Two Lengths



2. Solder the red wire attached to the Afterburner module to the middle pin of the potentiometer (denoted as pin 2 in Figure 5.51). It might help to “tin” the pins of the potentiometer by applying solder directly to them before soldering on the wire. This will give you a stronger solder joint.
3. Solder the red wire attached to the resistor to the bottom pin of the potentiometer (closest to the edge of the GBA case housing, denoted as pin 3 in Figure 5.51).

Figure 5.51 Close Up of Potentiometer with Pins Denoted

4. Your completed Afterburner installation (before you close up the case) should resemble the one in Figure 5.52.

Figure 5.52 Completed Afterburner Installation (With Brightness Control)

5. You can now reassemble your GBA as described in the previous “Opening the GBA Console” section. Take care to ensure that none of the wires are pinched or broken as you put the GBA back together.

Congratulations! The hack is complete and you should now have the Afterburner Internal Lighting Kit installed in your GBA. Turn on your system to make sure that the light is operational and that your brightness dial is functioning (if you installed one).

Figure 5.53 shows the Game Boy Advance in normal indoor lighting before and after the Afterburner kit was installed. Notice that difference!

Figure 5.53 GBA Before and After Afterburner Modification



Under the Hood: How the Hack Works

Triton Labs' Afterburner GBA Internal Lighting Kit comes with all the components necessary to create a professional-looking front light for the GBA's LCD panel. The core component to the kit is a custom-made front light module. When power is applied to the module, the LED bar at the bottom is illuminated and the special light-guide construction of the unit spreads the light evenly over the surface of the unit. During the hack, the front light is placed on top of the GBA's LCD, which is normally dim and hard to see in low lighting conditions. The front light will help illuminate the LCD, bringing out the sharp colors and allowing you to use your GBA even in complete darkness.

The optional adjustable brightness control is simply a variable resistor (also known as a *potentiometer*). Turning the potentiometer will change its resistance and vary the current being allowed to flow to the actual LEDs within the front light module. The less current that is allowed to flow, the dimmer the lights will be. The more current that is allowed to flow, the brighter the lights will be (see the “Electrical Engineering Basics” appendix for details on how resistors work). So, using the brightness control knob gives you a range from very dim to very bright, with many steps in between.

Enhancing Your Afterburner with the GBA Stealth Dimmer Chip

The GBA Stealth Dimmer Chip is a replacement to the analog brightness control wheel that comes with Triton Labs' Afterburner GBA Internal Lighting Kit. (See the previous “Light Up Your LCD with the GBA Afterburner” section.) By installing this chip in place of the potentiometer, you will have complete control of the Afterburner’s front light brightness using the GBA buttons. This hack makes for a much cleaner appearance because the installation of the chip is completely internal to the GBA, and you won’t have to modify your case to add the potentiometer.

Using the A, B, and Select buttons, simply select the light level that is suitable to your playing environment, and the chip will remember your settings. The Stealth Dimmer Chip provides the following features (obtained from www.division-6.com, the Web site of Division 6, the original creators of the Stealth Dimmer Chip):

- **Dimming** Smoothly scroll through many levels of brightness. Hold down the **Select** button and use **A** (brighter) or **B** (darker).
- **Instant On/Off** Hold **A** or **B** down first, then press **Select** to skip to full-on or full-off. You can also hold **A** or **B** during power-up.
- **Lock Feature** Press **Select** for 5 seconds to lock the chip so that it will ignore any button presses (Afterburner blinks to confirm). This is useful if the game you are playing uses the **Select + A** or **B** combination.
- **Memory** The chip will remember your dimmer settings if you turn the GBA off or even remove the battery.
- **Power-Save** Shuts off the Afterburner light if the **A**, **B**, or **Select** buttons are not pressed for 5 minutes. Comes back on at the touch of a button. Can be disabled by holding the **Select** button down for 5 seconds (Afterburner blinks to confirm).
- **Standby Sensing** If the GBA is in standby mode, the chip will not activate the power-save feature. This prevents you from having to wake the unit up and then turn the light back on.
- **Flicker-Free** Synchronizes the Afterburner with the GBA’s LCD for flicker-free dimming.
- **Hum-Free** Does not introduce a humming noise into the headphone output of the Game Boy Advance when dimming.
- **Built-In Diagnostics** Helps with installation and troubleshooting by blinking error codes on the Afterburner whenever a connection problem is detected.



Preparing for the Hack

This hack requires that your GBA already have the Afterburner GBA Internal Lighting Kit installed with or without the additional analog brightness control wheel. If you don't have the Afterburner installed, simply follow the installation instructions in the previous "Light Up Your LCD with the GBA Afterburner Mod" section of this chapter.

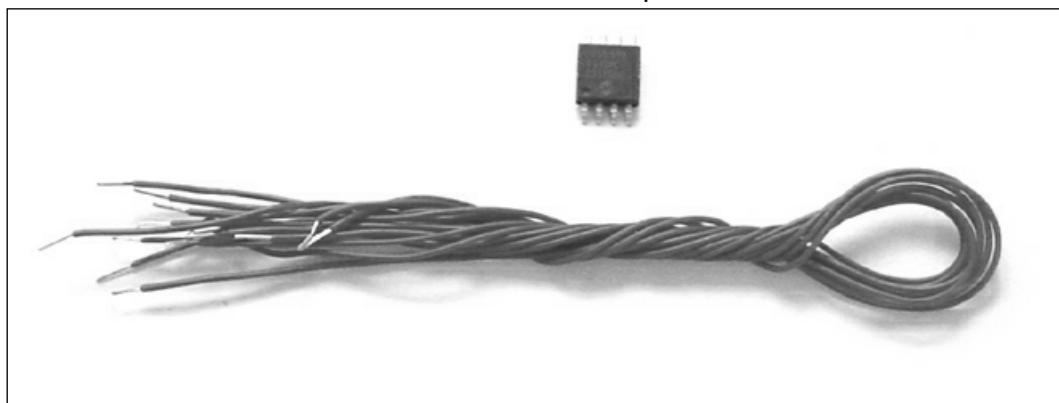
To complete this hack, you will need the following two items:

- Small piece of double-sided tape
- GBA Stealth Dimmer Chip V4 Kit

The GBA Stealth Dimmer Chip V4 Kit was created by Division 6 (www.division-6.com). Many clones and copycats of the Stealth Dimmer Chip are out there, especially from many online video game import companies. The only way to be sure that you are receiving an official Stealth Dimmer Chip is to order it directly from Division 6 or from one of its authorized distributors, such as Goldberg-Mods (www.goldberg-mods.co.uk). Beware that some of the clone vendors are calling their Stealth Dimmer Chip a V4 when, in fact, it is actually the pinout and functionality of the V2 or earlier V1 version (as we learned the hard way during the writing of this chapter). If you have an earlier version of the chip, you can find installation instructions at www.division-6.com/quick.htm and www.division-6.com/diagram.jpg.

The typical contents of the kit (see Figure 5.54) include the actual chip and a small bundle of connection wires. You can optionally choose to order the chip from Division 6 with no wires at all or with the wires already soldered to the device, which will save you some time during installation.

Figure 5.54 Contents of the GBA Stealth Dimmer Chip V4 Kit



You'll also need the following tools:

- Jeweler's size Phillips screwdriver (to open the GBA case)
- NES Tri-Wing security screwdriver (to open the GBA case)

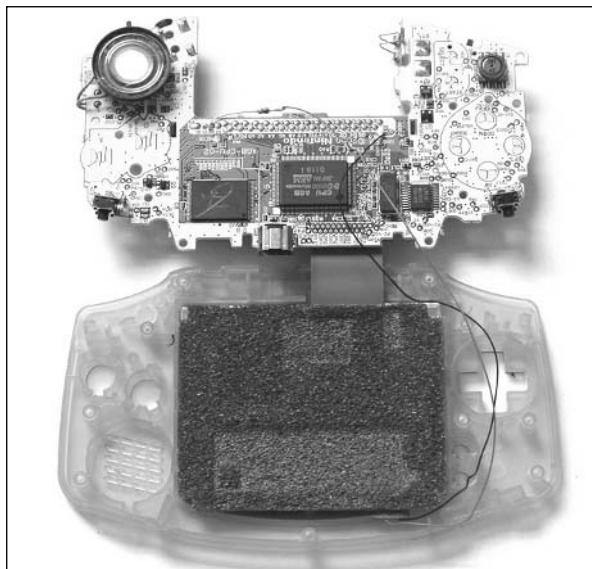
- Soldering iron and solder
- Wire strippers

Performing the Hack

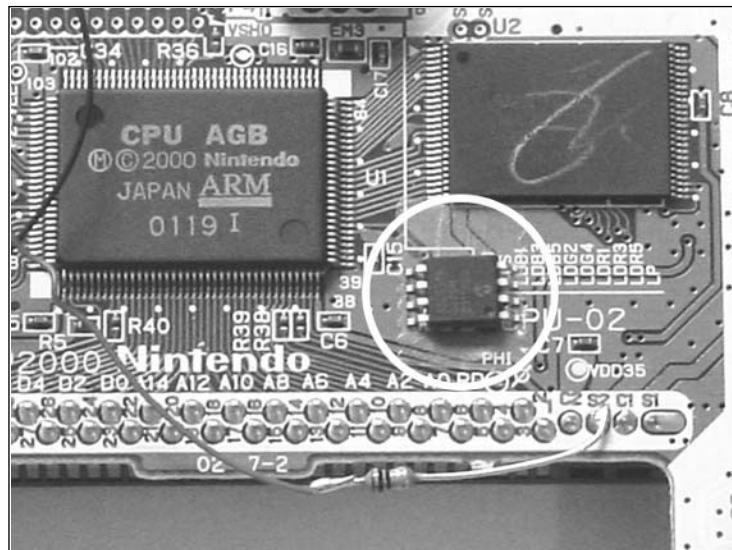
Perform the following:

1. Open your GBA as described in the “Opening the GBA Console” section earlier in this chapter. Skip Step 5 of the process, since we will be leaving the flexible LCD cable connected to the GBA circuit board for this hack. Your open unit, ready for the hack, should resemble the one shown in Figure 5.55.

Figure 5.55 GBA with Afterburner Opened and Ready for Modification



2. If a potentiometer is installed in your GBA for brightness control (optionally installed during the original Afterburner front light installation), disconnect all the wires going to it. Remove and discard the potentiometer, if you wish. Since the Stealth Dimmer Chip adjusts the brightness of the Afterburner front light using the GBA’s A, B, and Select keys, you won’t need it any more.
3. Using a small piece of double-sided tape, stick the Stealth Dimmer Chip to the GBA circuit board as shown in Figure 5.56. This will prevent the chip from moving around while you are attaching the wires to it. Ensure that you orient the chip as shown in the figure, with pin 1 located in the lower-right corner. It is recommended that you place the chip to the right of the CPU in the empty area where there are no components.

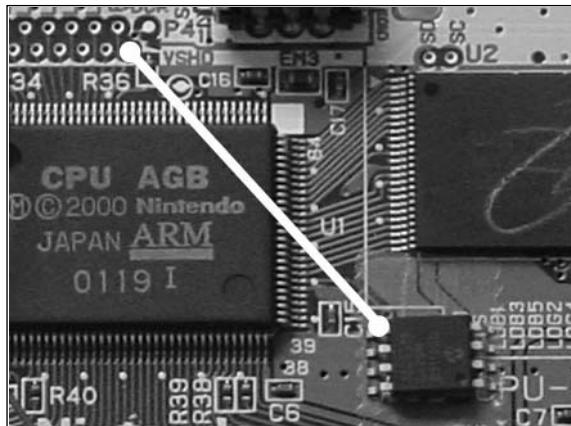
Figure 5.56 Tape the Stealth Dimmer Chip to the GBA

The hack is essentially completed by connecting the eight pins of the Stealth Dimmer Chip to eight locations on the GBA circuit board. The following steps will guide you through the process. See the “Under the Hood: How the Hack Works” section that follows for details on the functionality of the Stealth Dimmer Chip and what connections you are interfacing to on the GBA circuit board.

We will start on the upper-left corner of the chip (pin 5) and work counter-clockwise around the chip until all the wires are connected. You should cut off the excess wire length before making each connection to ensure that you don’t have a spaghetti-tangle of wires when the hack is complete.

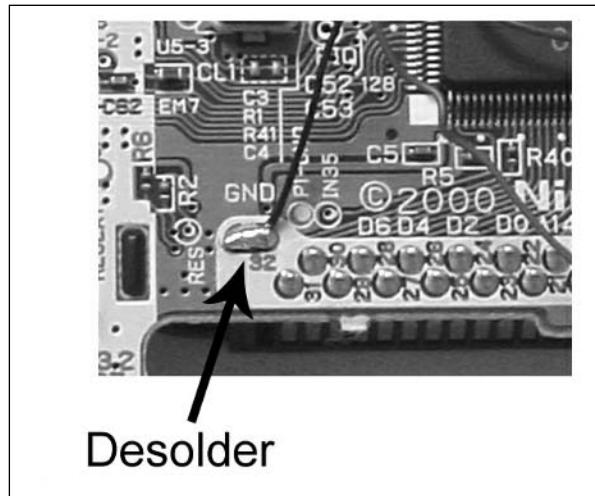
Perform the following:

1. Solder a wire from the testpoint near R36 (pointed to by the VSHD marking on the silkscreen) to pin 5 of the Stealth Dimmer Chip (see Figure 5.57).

Figure 5.57 Connect the Wire from VSHD to Stealth Dimmer Chip Pin 5

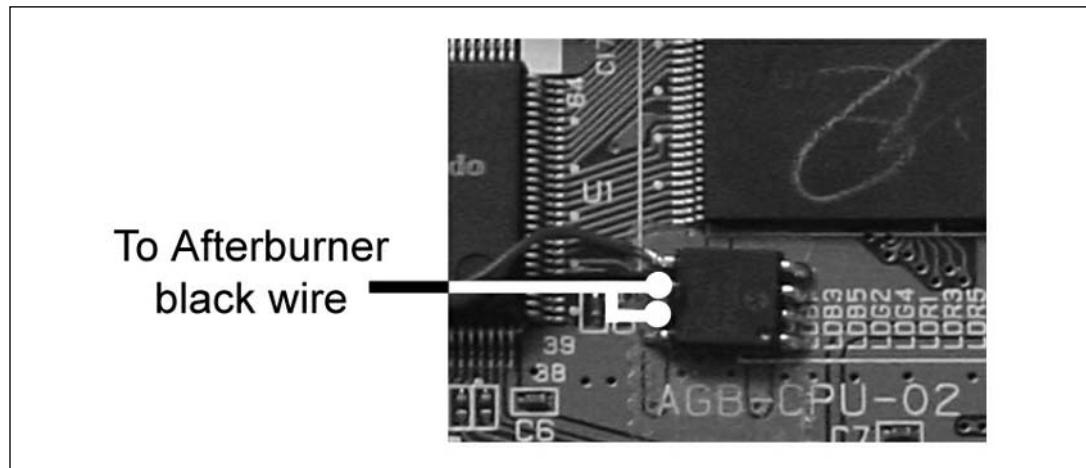
2. Remove the black wire that is connected from the Afterburner module to the GND connection on the GBA circuit board (see Figure 5.58).

Figure 5.58 Remove the Original Afterburner Connection

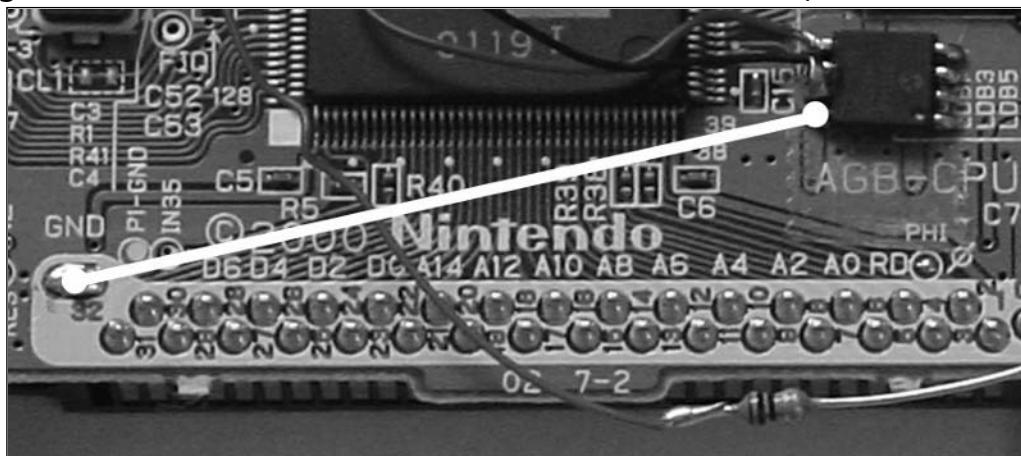


3. Connect the black wire that you just removed to pins 6 and 7 of the Stealth Dimmer Chip (see Figure 5.59).

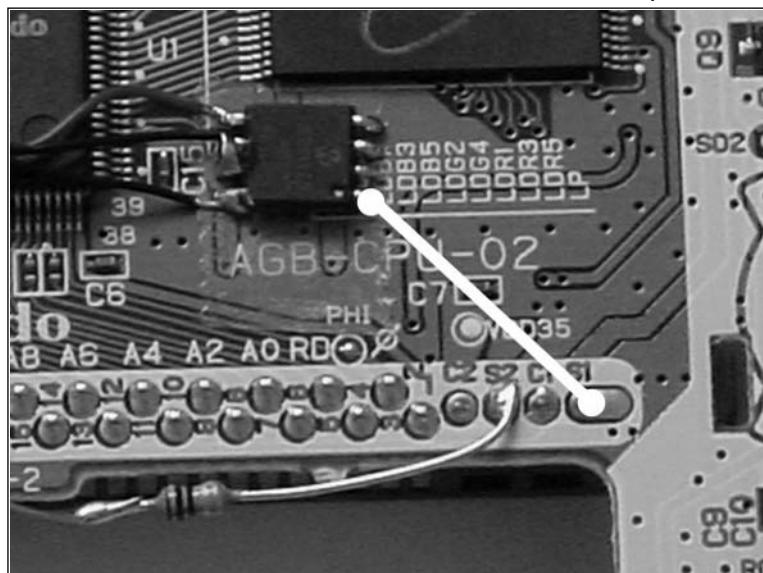
Figure 5.59 Connect the Black Wire from the Afterburner to Stealth Dimmer Chip Pins 6 and 7



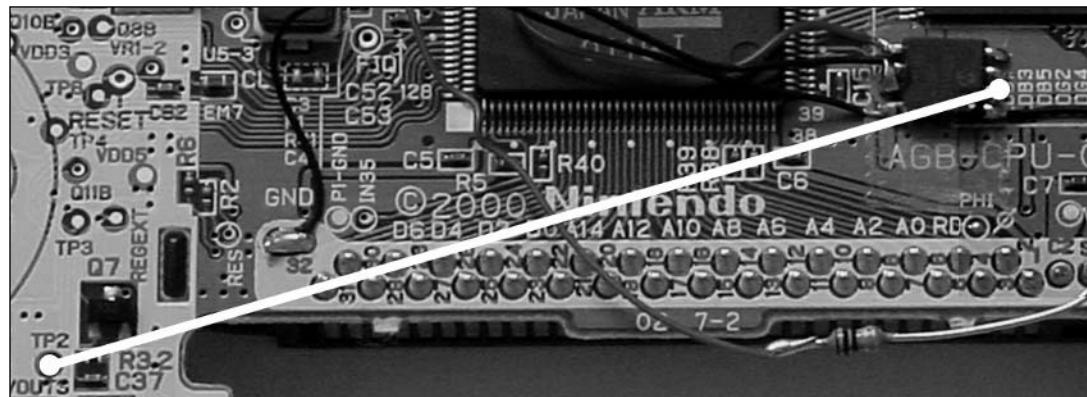
4. Solder a wire from the GND connection to pin 8 of the Stealth Dimmer Chip (see Figure 5.60).

Figure 5.60 Connect the Wire from GND to Stealth Dimmer Chip Pin 8

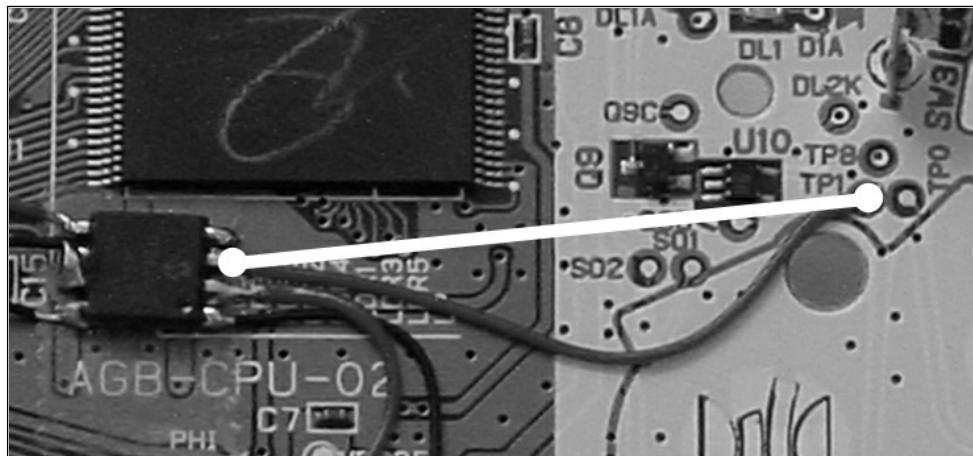
5. Solder a wire from the pad labeled S1 to pin 1 of the Stealth Dimmer Chip (see Figure 5.61).

Figure 5.61 Connect the Wire from S1 to Stealth Dimmer Chip Pin 1

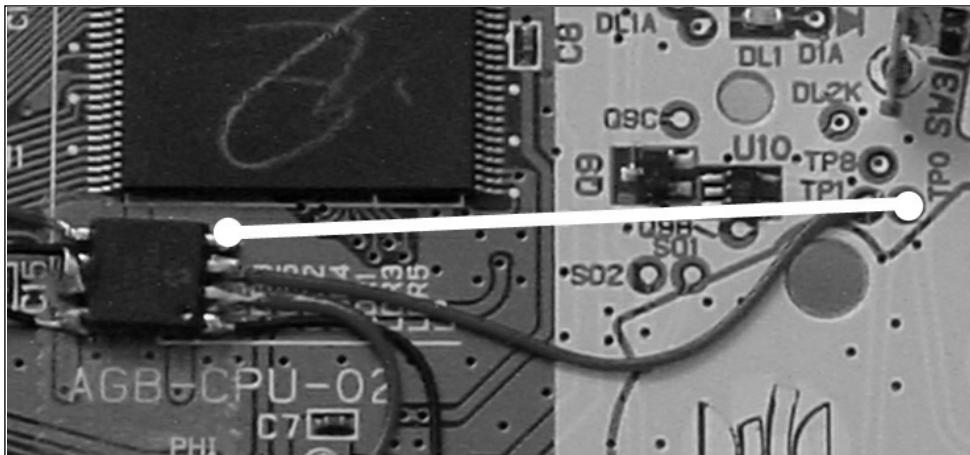
6. Solder a wire from the pad labeled TP2 to pin 2 of the Stealth Dimmer Chip (see Figure 5.62).

Figure 5.62 Connect the Wire from TP2 to Stealth Dimmer Chip Pin 2

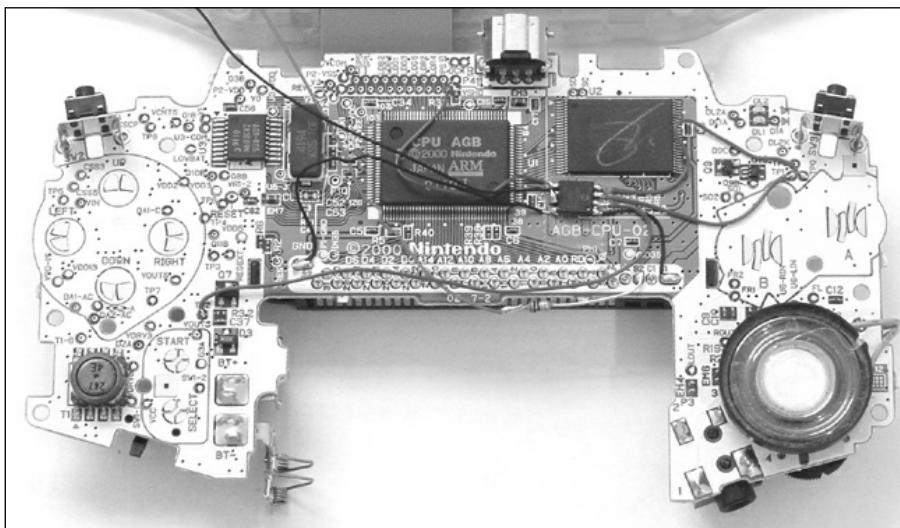
7. Solder a wire from the pad labeled TP1 to pin 3 of the Stealth Dimmer Chip (see Figure 5.63).

Figure 5.63 Connect the Wire from TP1 to Stealth Dimmer Chip Pin 3

8. Solder a wire from the pad labeled TP0 to pin 4 of the Stealth Dimmer Chip (see Figure 5.64).

Figure 5.64 Connect the Wire from TP0 to Stealth Dimmer Chip Pin 4

Your completed Stealth Dimmer Chip installation (before you close up the case) should resemble the one in Figure 5.65.

Figure 5.65 Completed Stealth Dimmer Chip Installation

9. You can now reassemble your GBA as described in the previous “Opening the GBA Console” section. Take care to ensure that none of the wires is pinched or broken as you put the GBA back together.

Congratulations! The hack is complete and you should have full brightness control of the Afterburner at your fingertips. The Stealth Dimmer Chip V4 is fairly easy to use and is operated as follows:

- **Dimming** Hold down the **Select** button, then press **A** to make the Afterburner brighter or **B** to make the Afterburner darker. Release the buttons when your desired brightness has been reached.
- **Instant On/Off** Hold down **A** for on or **B** for off and tap the **Select** button. The Afterburner will instantly turn on or off. You can also hold **A** or **B** as you power on the GBA to instantly turn the Afterburner on or off.
- **Stealth Chip Lock** To lock the chip so that it ignores all subsequent button presses, hold the **Select** button down for 5 seconds. The Afterburner will blink once to indicate that the chip is locked. To unlock, hold **Select** down for another 5 seconds (the Afterburner will blink twice). This feature is useful if you need to use the **Select-A** or **B** button combination in a game that you are playing and don't want the dimmer to respond.
- **Disable Auto Power-Save** By design, the Stealth Dimmer Chip will turn off the Afterburner after 5 minutes of inactivity. To disable this feature and prevent the dimmer from shutting off the Afterburner, hold the **Select**, **A**, and **B** buttons down simultaneously for 5 seconds. The Afterburner will blink once to let you know that the power-save feature has been disabled. To re-enable the power-save feature, hold **Select**, **A**, and **B** down for 5 seconds again. The Afterburner will blink twice.

Under the Hood: How the Hack Works

The GBA Stealth Dimmer Chip consists solely of a Microchip PIC12F629 processor. Although this general-purpose microprocessor is very small (only 8 pins), it is fully programmable and powerful enough for many applications. You can obtain the technical data sheet from <http://ww1.microchip.com/downloads/en/DeviceDoc/41190c.pdf>.

The Microchip PIC processor is programmed with custom code that will monitor the state of the buttons (which are used to enable, disable, and set the desired brightness of the Afterburner). Table 5.2 shows the pinout of the Stealth Dimmer Chip and the functionality of each pin.

Table 5.2 Stealth Dimmer Chip Pinout

Pin	Description
1	+3.3 volts (Chip Power)
2	"Enable" Input (Select)
3	"Darker" Input (B)
4	"Brighter" Input (A)
5	Sync Input

Continued

Table 5.2 Stealth Dimmer Chip Pinout

Pin	Description
6	AB Out (-)
7	AB Out (-)
8	Ground

When the correct button sequence is detected, the Stealth Dimmer Chip will vary the brightness of the Afterburner front light using a technique known as pulse width modulation (PWM). The ground connection of the Afterburner is fed into two general-purpose I/O pins of the PIC instead of going directly to ground. Each of these pins can sink up to 25mA of current to ground (as defined in the PIC12F629 data sheet) for a total of 50mA—well within the typical consumption of the Afterburner front light. When the pins are set low, current is allowed to flow through the front light, thus completing the electrical connection. When the pins are set high, no current will flow, because the potential difference between the positive (anode) and negative (cathode) pins of the front light is 0V.

By varying the relationship between the time that the pins are low (thus enabling the Afterburner) and the time that the pins are high (thus disabling the Afterburner), known as the duty cycle, the resulting brightness seen by the human eye will vary. A low duty cycle (for example, the pin is low for 10% of the time and high for 90% of the time) will result in a very dim light. A high duty cycle (in which the pin is low for more time than it is high) will result in a brighter light.

Nintendo GBA Technical Specifications

This section provides a detailed overview of the Nintendo GBA hardware. We'll divide the system into its major building blocks:

- The central processor (CPU)
- Memory architecture
- The graphic system
- The sound system

The Central Processor

The processor in the Game Boy Advance is based on the popular ARM7TDMI core. This is a 32-bit device with a three-stage pipeline, a hardware multiplier, and lots of registers—it is quite versatile. The following two reference books are highly recommended for further low-level detail:

- ***ARM Architecture Reference Manual***, second edition, edited by David Seal; Addison-Wesley, 2000, ISBN 0-201-73719-1. This definitive reference book contains detailed information about all versions of the ARM and Thumb instruction sets, the memory management and cache functions, and optimized code examples.

- **ARM System-on-Chip Architecture**, second edition, by Steve Furber; Addison-Wesley, 2000, ISBN 9-201-67519-6. This book is a practical look at the ARM core. It presents and discusses the major issues of system-on-chip design, including memory hierarchy, caches, memory management, on-chip buses, and debug and production tests. It also provides an overview of the ARM processor family, enabling the reader to decide which ARM is best for the job at hand.

The Advanced RISC Machine, or ARM, is a Reduced Instruction Set Computer (RISC) processor design. As with most RISC processors, the instruction set is very regular, meaning that there are few ways to encode an instruction. RISC design makes it very easy to build a fast and inexpensive CPU. It does not, however, lead to very compact code. In general, *code density*—the number of instructions per unit of memory—is lower in RISC designs than it is in Complex Instruction Set Computer (CISC) designs.

The designers of the ARM decided that they could create a denser instruction set—which they dubbed the *Thumb instruction set*—by cutting out some features and making the instruction encoding a little less regular. They did this in a way that allows the two instruction sets to work together. The CPU essentially converts Thumb instructions into their equivalent ARM instructions on the fly. Each Thumb instruction is 16 bits, whereas the ARM instructions are 32 bits. There is a performance benefit to using 16-bit instructions in a computer with 16-bit memory. True, there may be cases for using Thumb instructions to speed up parts of a game, but there are also cases for using regular ARM instructions as well.

NOTE

 With the ARM processor, a *word* is 32 bits, a *halfword* is 16 bits, and a *byte* is 8 bits.

Handy reference sheets for the ARM and Thumb can be obtained from <http://re-eject.gbadev.org/files/armref.pdf> and <http://re-eject.gbadev.org/files/ThumbRefV2-beta.pdf>, respectively.

CPU Registers

The ARM has 16 registers accessible at any time. A *register* is a physical component of the processor; each register holds 32 bits. One of these registers, called R15, is the program counter, which keeps track of the current instruction being executed. Some other registers are used only under certain operating conditions, such as R13 and R14 (usually the stack pointer and link register) that are used when interrupts occur.

The ARM Procedure Call Standard (APCS) defines a convention for compilers to use when calling functions. This is the way the C compiler calls functions (except under certain optimizing conditions). You don't have to follow this convention when you write your own assembly routines, but you won't be able to call those routines from C unless you do. Table 5.3 details the register uses for APCS. When one procedure calls another, there is an assumption that some of the registers will not

be changed by the called procedure. The called procedure must save and restore these values if it needs to change one of these registers.

Table 5.3 ARM Procedure Call Standard Definitions

Registers	Preserve	Usage
R0-R3	No	Used for passing parameters to functions. Any parameters that don't fit here get pushed onto the stack.
R4-R10	Yes	Used primarily for register variables. Registers R9 and R10 are also used for stack manipulation when switching modules, but you'll seldom, if ever, need to worry about this.
R11 (fp), see Note	Yes	Used as the frame pointer. This register is typically set and restored during the prologue/epilogue code since it is the pointer through which all local variables are accessed.
R12 (ip), see Note	No	Interlink pointer. For most GBA code, this is a scratch register.
R13 (sp), see Note	Yes	This is the stack pointer. This points to the last item pushed onto the stack. The stack is a “full descending” stack, meaning that the stack grows downward (toward lower addresses) in memory and the pointer always points to the next item to pop from the stack.
R14 (lr), see Note	No	This register, known as the link register, holds the return address for the subroutine. This register is often pushed on the stack and then popped directly into the program counter for the return.
R15 (pc), see Note	No	This register is the program counter. You directly change it only to execute a jump.

NOTE



These synonyms are often used in assembly (and disassembled) code.

Memory Architecture

You might be aware that your PC has three distinct types of memory: main RAM, which holds all the programs and data you’re actively working with (ignore virtual memory for now, since this is treated as main RAM even though it is located on the hard disk); the hard disk, which stores information for long periods of time; and display memory on your video card.

The Game Boy Advance has similar kinds of memory. Like the PC, each memory address of the GBA refers to a single 8-bit location (which means that it is “byte addressable”). Also like the PC, the ARM processor in the Game Boy Advance can access 8, 16, or 32 bits at a time. Table 5.4 lists the

types of memory accesses the memory allows and the wait states each access incurs. Details of each memory type can be found in the following subsections.

Table 5.4 Game Boy Advance Memory Specifications

Memory Type	Access Widths	Wait States	Comments
IWRAM	8, 16, 32	0	32KB “Internal” Working RAM. Typically used for fast scratchpad RAM and for time-critical code.
EWRAM	8, 16*	2	256KB “External” Working RAM. Typically used for main data storage and Multiboot code.
VRAM	8**, 16*	0	96KB Video RAM. Stores all graphics data. Can only write 16 bits at a time.
Game ROM	8, 16*	4/2 or 3/1	Part of the game cartridge, ROMs provide the game’s program and data and can be read in either slow (4/2) or fast (3/1) mode.
Game Save Memory	8, 16*	Special	An optional part of the game cartridge, the Game Save Memory is typically a non-volatile area used for saving the state, score, or other data of the particular game.

* The CPU can access these memories with 32-bit reads and writes performed as two consecutive 16-bit accesses.

** VRAM can be read as bytes. Writing a byte causes both bytes in the addressed 16-bit halfword to be written with the same value.

NOTE



The CPU reads memory by sending an address to the memory at the start of a cycle and then reading the data at the end of that cycle. Ideally, the memory is fast enough to provide the data that quickly. Slower memory tells the CPU to wait for one or more additional cycles before reading the data. These extra cycles are called *wait states*. The fastest memories, such as IWRAM and VRAM, have no wait states (abbreviated OWS) and therefore return data in a single cycle. EWRAM is 2WS memory, returning data in three cycles—one normal cycle plus two wait states.

Internal Working RAM

Internal Working RAM (IWRAM) is the only memory directly accessible on the 32-bit internal data bus of the CPU core because it is actually built into the CPU itself. This is why it’s called *Internal*

WRAM as opposed to External WRAM (covered in the next subsection). IWRAM is the fastest memory in the Game Boy Advance and is also the only memory that can be accessed 32 bits at a time. The speed and width of this memory make it ideal for running ARM code at full speed. Unfortunately, there are only 256 Kbits (32KB) of this memory.

External Working RAM

One might think something named External Working RAM (EWRAM) would be located external to the actual product (such as on a removable memory card). However, in the case of the GBA, the RAM is denoted as “external” because it sits outside the CPU’s core on the 16-bit data bus. EWRAM is where you will store large data items, since 2,048 Kbits (256KB) are available. However, each memory access takes three cycles—much slower than IWRAM. You will probably want to cache graphics here before transferring them to VRAM for display. You can also place programs here using the Multiboot protocol. (See the “Homebrew Game Development” section in this chapter for more details.)

Graphics Memory

Three sections of memory deal exclusively with video and the display screen: video memory, palette memory, and object attribute memory.

Video RAM

Video memory is where all graphics data must be stored for display on the screen. Video RAM (VRAM) is zero wait-state memory, like IWRAM, but sits on the 16-bit data bus, so you can only move data half as fast as in IWRAM. How VRAM is used depends greatly on the video mode and other features that your program selects.

Care must be taken when writing to VRAM during the time when the screen is being drawn. Attempting to change memory that is being used to draw the screen can result in graphics glitches and image tearing (an event in which the image is being drawn while the screen is also being refreshed, resulting in an uneven image that looks like it is being torn apart). Furthermore, VRAM accesses during screen time can be delayed while the CPU waits for the video hardware to perform its accesses.

Palette Memory

Most of the Game Boy Advance’s video modes use palettes to specify the colors being used. The Game Boy Advance has two separate 256-color palettes: one for background images and one for sprites. Each of these palettes is further divided into 16 palettes of 16 colors in some modes, allowing graphics data to be compacted even more. Color 0 of any palette is defined as transparent no matter what value is actually stored in the palette memory. Palettes are usually updated during the vertical blanking (VBlank) period, at which time the screen is not being drawn. Video mode 4 doesn’t use a palette at all but instead directly addresses colors in each pixel. (See “The Graphic System” subsection for more details.)

Object Attribute Memory

Object Attribute Memory (OAM) is where you store the attributes, or descriptions, of what, how, and where a sprite is to display. The actual graphics data comes from VRAM, but the sprite's position, size, and other information come from the OAM. OAM is commonly updated during Vblank, and one often mirrors this data set in main memory for improved speed.

Game ROM and Game Save Memory

The GBA runs a game from an external game cartridge that is plugged into its cartridge connector. The connector provides access to the internal ARM processor signals and can be used to interface with external memory and other devices.

The GBA game cartridge contains the game's program and data stored in Read-Only Memory (ROM). Other cartridges, known as *Flash cartridges*, contain Flash-based, nonvolatile memory and can be programmed (and reprogrammed) at will with your own data using a Flash Linker device. Some cartridges also contain additional memory for saving the state of games using battery-backed Static RAM (SRAM), Flash ROM, or Serial Electrically Erasable Programmable Read-Only Memory (EEPROM).

Like EWRAM, the game cartridge ROM is accessible via the 16-bit data bus. There are two speeds at which ROMs can operate and two modes in which they can be accessed. Speeds for the ROMs are given as a pair of wait-state values, such as 3/1. The overriding factor is whether each access to the ROM can be classified as sequential or nonsequential.

A *nonsequential access* occurs whenever a new area of the ROM is read. Sending the memory address to the ROM takes extra time, and these accesses take the number of wait states indicated by the first number. In this example, the nonsequential access will take four cycles (three wait states plus the normal cycle).

A *sequential access* occurs when the very next access to the ROM is at the next address. In this case, the ROM already has the next address available and only takes the smaller number of wait states. This use of sequential accesses means that a consecutive sequence of instructions that have no other data accesses can run with only one wait state for faster ROMs. Even slower ROMs (running with 4/2 wait states) can equal EWRAM speed for these short bursts, while fast ROMs can outpace EWRAM during such a run. What this means, basically, is that a game cartridge is capable of slow-mode and fast-mode access.

The Graphics System

Arguably the most important aspect of a video game console (aside from the quality of the games written for it) is the graphics system. The Game Boy Advance has an intriguing selection of video modes from which to choose, giving game developers many options and capabilities. There are three tile-based modes that resemble the previous Game Boy graphics systems. In addition, there are three new bitmap-based modes that provide more creative freedom with a game.

The GBA requires exactly 280,896 clock cycles to display the entire video buffer to the screen. This equates to a refresh rate of 59.73 Hz (cycles per second). Therefore, the maximum frame rate on the Game Boy Advance is approximately 60 frames per second.

Tile-Based Modes (0–2)

The GBA provides three tile-based modes. A *tile* is a small 8-pixel-by-8-pixel section of the screen.

Mode 0

In Mode 0, four text background layers can be shown. Backgrounds 0–3 all count as text backgrounds and cannot be scaled or rotated.

Mode 1

Mode 1 is similar in most respects to Mode 0, the main difference being that only three backgrounds are accessible—0, 1, and 2. Backgrounds 0 and 1 are text backgrounds; background 2 is a rotation/scaling background.

Mode 2

Like Modes 0 and 1, Mode 2 uses tiled backgrounds. It uses backgrounds 2 and 3, both of which are rotation/scaling backgrounds.

Bitmap-Based Modes (3–5)

The GBA provides three bitmap-based modes. Each of these video modes has a defined resolution and video buffer.

Mode 3

Mode 3 is a standard 16-bit (nonpalettued) mode at a resolution of 240 x 160. The map starts at 0x06000000 and is 76,800 bytes long. This mode allows the full color range to be displayed at once. Unfortunately, the frame buffer in this mode is too large for page flipping (a method of reducing flicker on the screen). One option to get around this is to copy a frame buffer from work RAM into VRAM during the retrace.

Mode 4

Mode 4 is an 8-bit (palettued) mode at a resolution of 240 x 160. Bit 4 of the REG_DISPCNT register sets the start of the frame buffer to 0x06000000 when it is zero and 0x600A000 when it is one. The palette is at 0x5000000 and contains 256 16-bit color entries. Swapping the bitmap by flipping bit 4 and drawing in the one that isn't displayed allows for page-flipping techniques to be used.

Mode 5

Mode 5 is another 16-bit mode at a smaller resolution of 160 x 128. The display starts at the upper-left corner of the screen but can be shifted using the rotation and scaling registers for background 2. The advantage of using this mode is that two frame buffers are available to be used to perform page-flipping effects. Bit 4 of the REG_DISPCNT register sets the start of the frame buffer to 0x06000000 when it is zero and 0x600A000 when it is one.

The Sound System

The sound system in the Game Boy Advance comprises four frequency modulation (FM) synthesis channels for generating sound effects and music, which were used on previous Game Boy models, and two 8-bit pulse width modulation (PWM) channels that act as digital-to-analog converters (DACs) for playing digital sound effects and music. The two channels, called Direct Sound A and Direct Sound B, are capable of playing back 8-bit signed pulse code modulation (PCM) samples. PCM is a raw format that is supported by most sound editor programs.

There is no built-in sound mixer for *synchronous playback*, so programmers must write their own mixers or use a third-party library. A *mixer* allows multiple sounds to be played at the same time. Conceptually, it does this by “mixing” them together. Without a mixer, it is only possible to play one sound at a time, which is called *asynchronous playback*. Most developers use a digital sound library with a real-time mixer for playing true digital sounds on the GBA, such as Krawall GBA (<http://synk.at/krawall>).

Homebrew Game Development

There are many hardware and software resources available to enable you to write your own Game Boy Advance games. Even though hobbyists write such games in their spare time, many of the games rival the quality of those that are commercially available. Hundreds of homebrew applications have been created and are available for free.

As discussed earlier, the GBA is based on the popular ARM7 microprocessor core. The ARM7 is used in dozens of consumer electronics products including VCRs, DVD players, TVs, satellite receivers, MP3 players, DSL and cable modems, Pocket PCs, GPS transceivers, and video game consoles. The result of the ARM7's popularity is that assemblers and compilers are available for free, and even ARM, Ltd., the creator of the ARM processor, has released development tools into the public domain, going by the premise that the more tools that are available, the more applications and devices will use the processor. A search on the Web for "Game Boy Development" or "ARM7 Development" should yield a number of useful resources for aspiring homebrew game programmers.

The Multiboot cable (also referred to as the Multi Boot Version 2 or MBV2; see Figure 5.66) allows you to run programs directly on the GBA through its multiplayer EXT connector. During normal operation, when the GBA detects that a multiplayer cable is connected, the GBA will attempt to download a small binary program into memory from the host GBA (which is running the host game—for instance, Mario Kart Super Circuit). Games with multiplayer capability allow up to four players to participate in a game, although only one of the players is using the actual game cartridge.

Figure 5.66 The Multi Boot Version 2

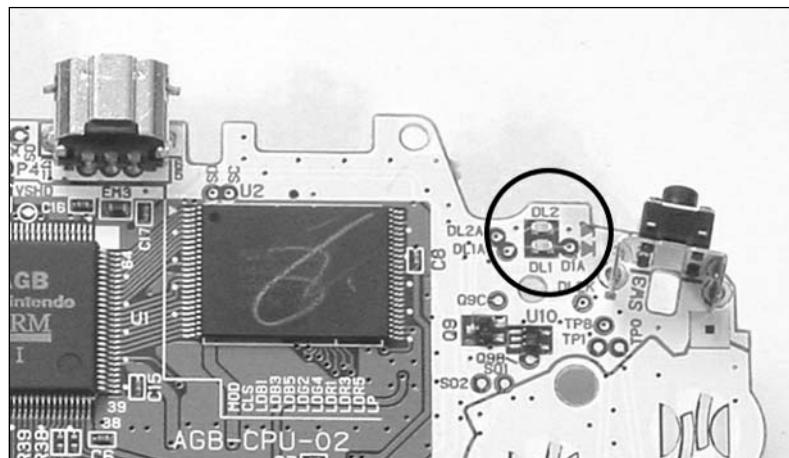
The MBV2 essentially fools the GBA into thinking it is linking up to a multiplayer game, when in fact it is connecting to the host PC. The MBV2 software running on the PC handles all the communications to the GBA using the multiplayer link protocol (a nonstandard 16-bit asynchronous interface) and can transfer game binaries directly into the EWRAM of the GBA. The GBA can only support games up to 256KB in size loaded via the Multiboot method (which is the maximum size of the EWRAM).

Other Hacks

The hacks presented in this chapter are only the beginning of what you can do with your GBA. We've listed some additional hacks here:

- **GBA Power LED Modification** This is a classic beginner hack suitable for any system, and the GBA is no exception. When powered on, an LED on the front panel of the GBA illuminates green. It is possible to replace the LEDs inside the GBA to another color, thus replacing the boring, standard green with something more unique, like blue or purple. Examples of other LED modifications can be found in the Nintendo NES, Atari 5200, and Atari 7800 chapters in this book.

On the front of the GBA circuit board are two green LEDs denoted by DL1 and DL2 (see Figure 5.67). These small surface-mount devices can be replaced with another LED color of your choice. When replacing the LEDs, be sure to also replace the associated current-limiting resistors on the other side of the board (most likely they are the two denoted by R10 and R25) so you do not damage your new LEDs and so you will have the desired brightness.

Figure 5.67 Power Indicator LEDs on the GBA Circuit Board

- **Loading UNIX onto the GBA** A popular hack with most any game console, porting a UNIX-variant (such as Linux) to the GBA has been an ongoing challenge for many developers. An excellent report is available from www.kernelthread.com/publications/gbaunix, in which the author details his successful attempts at loading the fifth edition (June 1974) of UNIX onto a GBA. The report contains a nice technical description of the GBA hardware, basic UNIX concepts, and demonstrations and downloads of his gbaunix tool.
- **Create your own e-Reader codes** The Nintendo e-Reader (see Figure 5.68) is a novel device that can optically read special barcodes printed on cards (containing a game's program code) and load the data into the GBA. Recent research, found at <http://users.skynet.be/firefly/gba/e-reader/index.htm>, has uncovered how the data is stored on the special barcodes, and tools have been written to allow you to create your own barcodes! Printing cards containing your homebrew game or demo is a much lower-cost and easier solution than creating custom GBA cartridges. Just for fun, Figures 5.69 and 5.70 show the circuitry inside the e-Reader device.

Figure 5.68 Nintendo e-Reader



Figure 5.69 Inside the Nintendo e-Reader, Front of Circuit Board

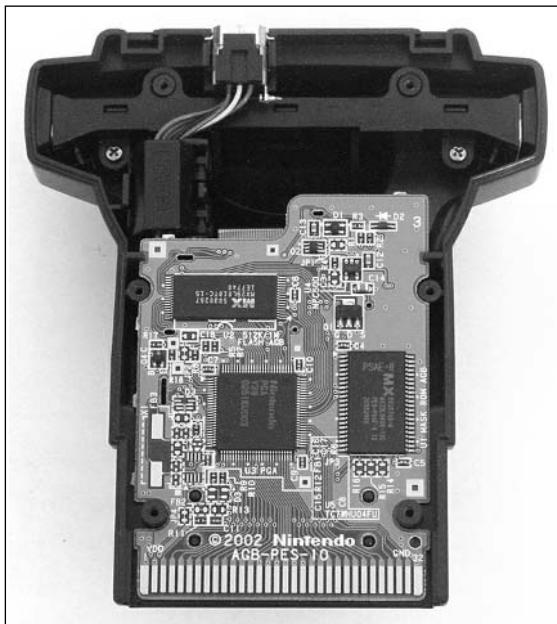
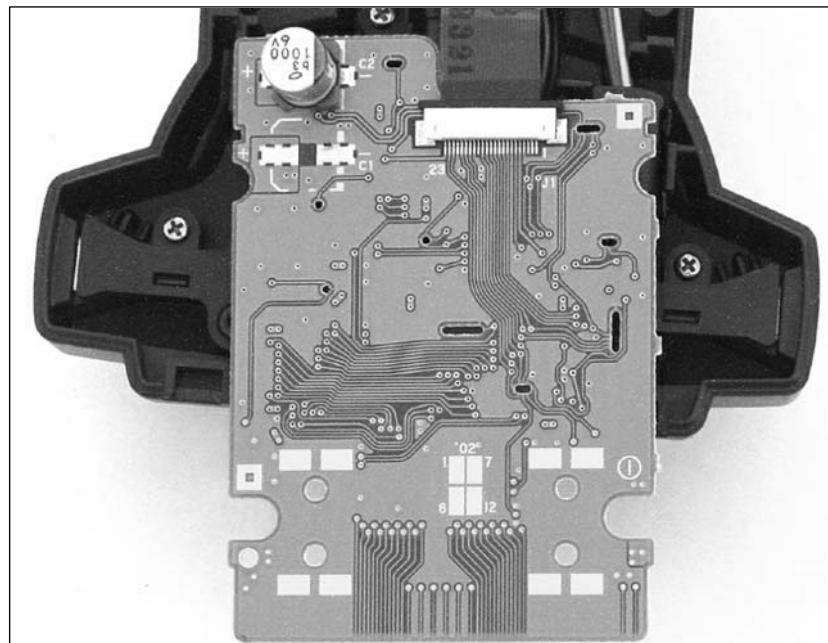


Figure 5.70 Inside the Nintendo e-Reader, Back of Circuit Board

- **Adding a custom skin** Though not exactly a hack, adding a custom case skin gives your GBA a unique personal flavor. Different themed and colored skins are available from many online video game retail stores and are simply attached to the front of your GBA housing like a sticker (see Figures 5.71 and 5.72). The nature of the adhesive allows the skin to be removed at will to restore the GBA case to its original configuration. However, the adhesive sticks well enough to not fall off during use.

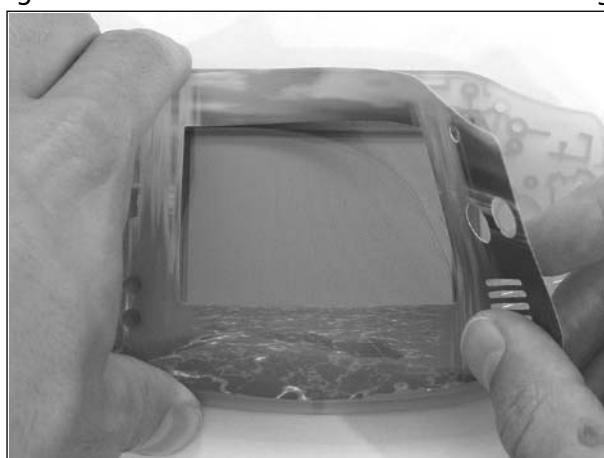
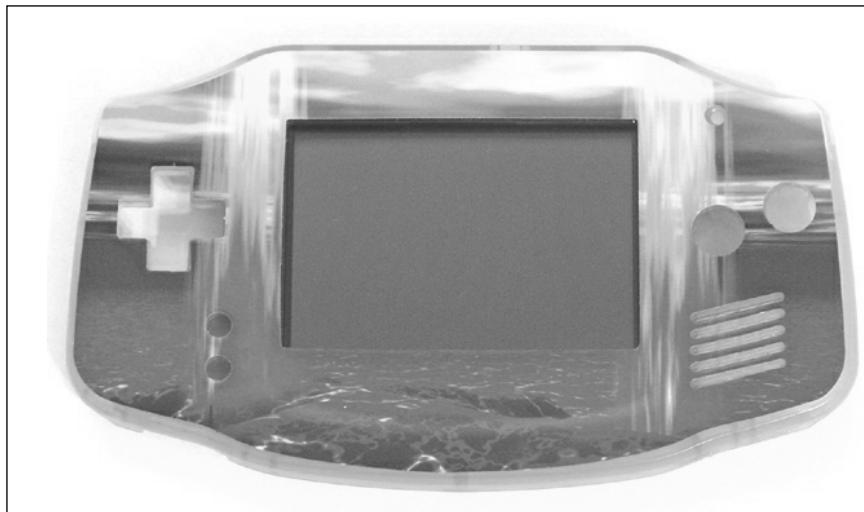
Figure 5.71 Adding the New Case Skin to the GBA Front Housing

Figure 5.72 GBA with New Case Skin

- **Turn your portable GBA into a full-fledged console** This project, created by Konstantin Khanyants, aims to convert the portable, standard GBA into a set-top video game console complete with an external monitor and support for real game controllers. Plenty of pictures, schematics, and information are available for you to build your own similar unit. You can find more information at <http://pages.prodigy.net/tan.ax/GBAConsole/gbaconsole.htm>.

Nintendo GBA Resources on the Web

The Web has a great number of resources for enthusiasts of the GBA. Whether you're looking for information about your favorite games, how to hack your handheld to give it capabilities never intended by the designers, or forums where you can discuss the GBA with other fans, you're sure to find it online somewhere. The following is a list of some of our favorite GBA-related sites:

- **Nintendo Game Boy Advance, www.gameboyadvance.com** Nintendo's official Game Boy Advance and Advance SP Web site. Features game information, news, and multi-media downloads.
- **Nintendo Europe, www.nintendo-europe.com/gameboyadvance** Nintendo's official Game Boy Advance Web site for European countries. Available languages include English, Dutch, Spanish, French, and Italian. Contains news, game information, system specifications, and active discussion forums.
- **GameSpy GBA Forums, www.forumplanet.com/gamespy/gba** Public forums related to the general discussion of the Game Boy Advance.

- **IGN Game Boy, <http://gameboy.ign.com>** A comprehensive GBA-related Web site containing articles, news, reviews, previews, features, and very active discussion forums.
- **GBAX.com, www.gbax.com** Excellent site and store devoted to providing products and hacks for the Nintendo GBA, GP32, Zodiac, and other portable consoles. Also features an often-updated news section for the latest information on available tools and software. Based in the United Kingdom.

Gamepark 32 (GP32)

Topics in this Chapter:

- Out of the Box: Configuring Your GP32
- Opening the GP32 Console
- Replacing the GP32 Screen Cover
- Repairing Your Buttons
- Accelerating Your GP32 (CPU Core Voltage Increase)
- Creating a DC Power Adapter
- Installing the Multifirmware Loader
- Homebrew Game Development
- Other Hacks
- GP32 Resources on the Web

Introduction

Developed by a relatively small and unknown Korean company called Gamepark, the GP32 (see Figures 6.1, 6.2, and 6.3) is a handheld system that seems to have been created with hackers and homebrew game developers in mind. Compared to many game console vendors that frown on people experimenting with and writing homebrew games for their machines, Gamepark has created an open platform with a welcoming development community.

Gamepark was founded in 1996, and the GP32 has been available in Korea since 2000. It has since made its way into other countries through such online retailers as Lik Sang (www.lik-sang.com) and GBAX.com. The GP32 currently retails for approximately US\$169, and it is estimated that 150,000 units have been sold. A number of commercial games exist for the system, though the quality and game play have had mixed reviews. Outside Korea, the GP32 has more of a cult following, which is part of the appeal for hacking it.

Figure 6.1 The GP32 Console, Front

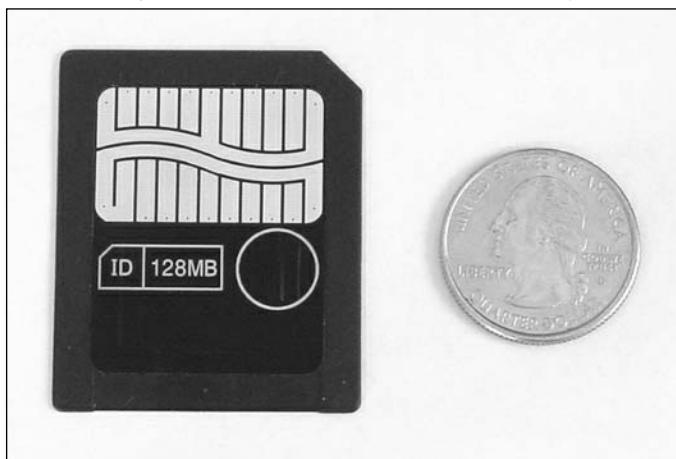


Figure 6.2 The GP32 Console, Back



Figure 6.3 The GP32 Console, Box

The system is arguably more powerful than the Nintendo Game Boy Advance (hacks for the GBA are discussed in Chapter 5) and is loved by a niche community for its capabilities in emulating other videogame systems. The handheld GP32 features a 3.5-inch, 320-by-240 pixel reflective Thin-Film Transistor (TFT) color LCD screen that can display up to 65,536 synchronous colors. The system is built around a 133 MHz, 32-bit ARM CPU, 512 KB of ROM, and 8 MB of RAM. The GP32 has decent audio capabilities for a portable platform, consisting of 16-bit PCM, four-channel stereo sound, and native support of MP3 decoding. For its input interface, the GP32 sports an eight-direction joystick (up, down, left, right, and diagonals) plus six buttons (A, B, Start, Select, Left, and Right). Games, applications, and data are stored on removable SmartMedia (SMC) memory cards (see Figure 6.4). Two AA batteries provide approximately 12 hours of operating time, and a DC adapter can be used to power the unit from a wall outlet, removing the need for batteries. (See the “Creating a DC Power Adapter” hack in this chapter for information on modifying an existing 3V DC power supply to be used with the GP32.) Through its external expansion port (labeled EXT on the side of the device), an optional 900 MHz RF adapter provides wireless, multiplayer, head-to-head support for certain games. The link cable to interface the GP32 to the PC connects via a mini-USB port.

Figure 6.4 The GP32 Cartridge, a SmartMedia External Memory Card

In late 2003 and again in March 2004, the European release of the GP32 was cancelled, supposedly related to Gamepark's unstable financial situation. A U.S. release of the system was also planned, but as this book goes to press, the current status is unknown. On a positive note, the Gamepark GP32 BLU (Backlight Unit) has recently been released and has sold unexpectedly well in Korea, giving the GP32 a new lease on life. The BLU version of the GP32 has a slightly different physical design and comes with a factory-installed backlit LCD, as opposed to the nonbacklit or modified Frontlight Unit (FLU). The FLU version seems to have problems with dust building up underneath the plastic protective LCD cover. This is most likely due to the fact that the FLU was added to the GP32 after the fact by a third-party and the modification was performed in an unclean environment.

Most of the GP32's homebrew development community seems to be focused on creating emulators for the system, so if you're a gamer hoping to have a portable emulation console, you're in luck. Videogame emulators exist for arcade games (using MAME, the Multiple Arcade Machine Emulator) and just about every game system, including the Atari 2600, Atari 5200, NES, Super NES, Nintendo Game Boy and Game Boy Advance, Sega Genesis, NEC TurboGrafx-16, NEC PC Engine, Commodore C64, Atari ST, and Sinclair ZX Spectrum (to name a few). More information can be found in the "Homebrew Game Development" section at the end of this chapter.

All in all, the GP32 is a powerful and exciting system. Getting started might not be the easiest thing (see the following "Out of the Box: Configuring Your GP32" section), but once you're up and running, the game console rivals the more popular handhelds and is much more flexible. Consider the GP32 a fledgling game system with decent hardware and a huge underground game development community, open to all. If you're a tinkerer or ever wanted to make your own game, this is one system that won't let you down.

The hacks in this chapter focus on the original GP32 (non-FLU) unit. The newer FLU and BLU units are similar in design and appearance, so most of these hacks should work just fine with them. However, be sure to read each hack fully before trying it on any version other than the original GP32.

Out of the Box: Configuring Your GP32

Out of the box, getting your GP32 up and running is not a task for the faint of heart. Essentially, each GP32 console has its own unique product ID that needs to be registered with Gamepark before applications can be executed. The registration process was intended as a form of copyright protection (which has actually been broken by a number of curious individuals), but it isn't user friendly. The good thing is that you only have to go through the registration process once!

The following steps guide you through the process of setting up your GP32 for the first time. This isn't the most elegant solution, but it is representative of the hacking necessary for all aspects of this system. Your situation might vary slightly, depending on your model type (BLU units do not require registration) or where you purchased the GP32 (some vendors register and preload the device with applications before sending it to the buyer). If your unit has not been preconfigured, follow these steps:

1. Register your device at www.gp32eu.com/register.php (optional). Simply enter your information into the forms and press the **Register Me** button. Your GP32 serial number (product ID) is located on a sticker underneath the batteries on the back of the device. It should be 13 characters long and will look something like the one shown in Figure 6.5.

The original method of GP32 registration was done on Gamepark's main Web site. English registration was available at <http://english.gamepark.com>, but that server has since disappeared. By entering your information into the registration form, you would have been prompted to download personalized versions of the required applications. However, since registration has been unreliable at best, even with the site listed here, we won't actually be relying on Gamepark for the applications, so this registration step is optional.

Figure 6.5 GP32 Product ID Sticker Behind the Battery Cover

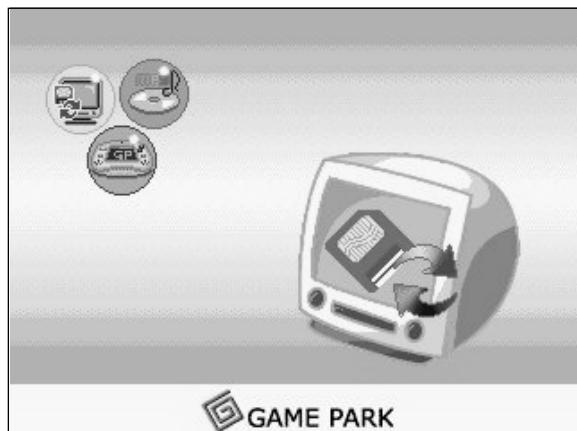


2. Install two AA batteries (or plug in a modified 3V DC adapter as discussed in the “Creating a DC Power Adapter” hack in this chapter).
3. Install a blank SMC card; 128MB is recommended so you’ll have plenty of space to store game files and MP3s.
4. From www.gp32eu.com/gputilities.php, download the following files:
 - GP32 USB drivers for your operating system
 - GP32 link program for your operating system, which lets you load data onto the GP32’s SMC card (PC-Link for Windows, GPlink for Linux, or Maclink for Mac OS)
5. Connect the GP32 to your computer with the supplied USB link cable and turn it on.
6. When prompted, install the GP32 USB drivers. With the USB drivers installed, the GP32 should be able to communicate with the PC. (We’ll verify this in a later step.)

To be able to run Free Executable (FEX) files, the primary file format used to load homebrew games onto the GP32, you must install a personalized version of Gamepark’s Free Launcher application that is tied to your specific product ID. The GP32 was designed to run only encrypted applications, so the personalized version of Free Launcher is necessary. Free Launcher is one of the applications provided by Gamepark during registration, but since we’re skipping the formal registration process, we’ll use some freely available tools offered by GP32 homebrew developers instead. Sound confusing? It is. But fear not, the next steps will guide you through the process.

7. Download Dark Fader’s GPS2 tool from <http://darkfader.net/gp32/files/GPS2.zip>.
8. On the GP32, select the **PC LINK** option (from the Main menu, press the **Select** button until you see an image of a computer and SMC card, then press **Start**; see Figure 6.6). Your GP32 will enter a new screen and should respond with a “Now waiting...” message while it waits for a host program on the PC with which to communicate.

Figure 6.6 GP32 PC LINK Option



- On your PC, run the **GPS2** tool and click the **Read** button. The fields of the dialog box will fill with your GP32 specific information: SMCID, PDUID, User, and Password (see Figure 6.7). For now, we are only concerned with the PDUID.

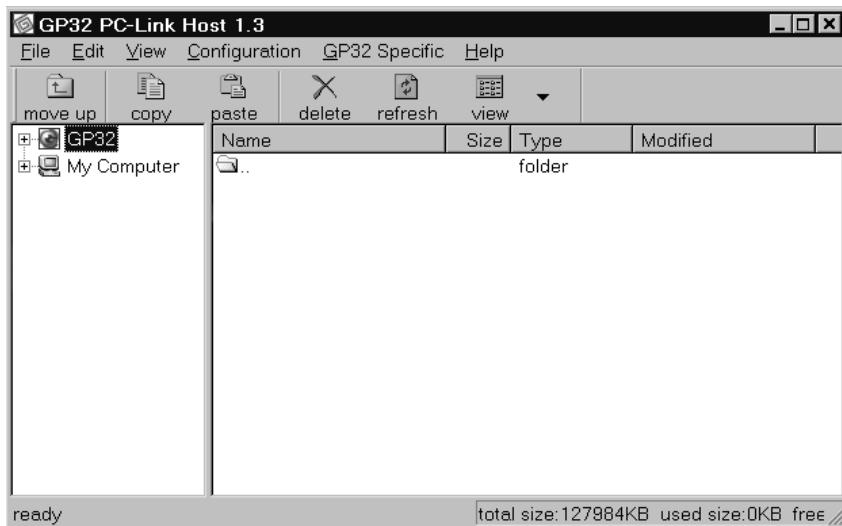
Figure 6.7 GPS2 Showing the Unique PDUID of the GP32



- Using an IRC (Internet Relay Chat) client, connect to an EFNet server (<irc.efnet.org>) and join the #gp32 channel. Popular IRC clients include mIRC for Windows (www.mirc.com), BitchX for Unix (www.bitchx.org), and Conversation for Mac OS X (<http://homepage.mac.com/philrobin/conversation>). Information about IRC can be found at www.irc.org.
- In the channel, type **!encrypt <PDUID> free_launcher** where <PDUID> is the PDUID displayed in the GPS2 tool. For example, for the GP32 device shown in Figure 6.7, you would type **!encrypt acbeb1b8b7a6aab4acbeb1b8b7a6aab4 free_launcher**. A private message will be sent to you from the GP32DEV user, similar to this:

```
<GP32DEV> Wait for few seconds and then download from:  
<GP32DEV> http://194.137.29.244/~gp32/acbeb1b8b7a6aab4acbeb1b8b7a6aab4.zip  
<GP32DEV> The zip archive contains two files: .gxc and .gxe files.  
<GP32DEV> Extract the contents of the zip archive under GP:\GAME in your SMC.  
<GP32DEV> The path structure in the zip archive must be preserved.
```

- Download the personalized Free Launcher application from the provided link and extract it to a temporary folder on your PC. Be sure to maintain the path structure of the archive.
- Install and run the GP32 link program that you downloaded earlier. For this example, we'll be using PC-Link Host for Windows. You will be greeted with a screen similar to the one shown in Figure 6.8.

Figure 6.8 GP32 PC-Link Host

14. Initialize your SMC card by choosing **File | Format**. This command will erase all data from the SMC card and create the GP32 file structure. Note that the SMC is formatted with a FAT12 file system, meaning that Windows long filenames are not supported. You can't use special characters, symbols, or spaces in the filenames, and files have a maximum 8.3 format. When the formatting is complete, you will see a listing of directories now existing on the GP32: GAME, MP3, GPSYS, GPMM, and GPETC.

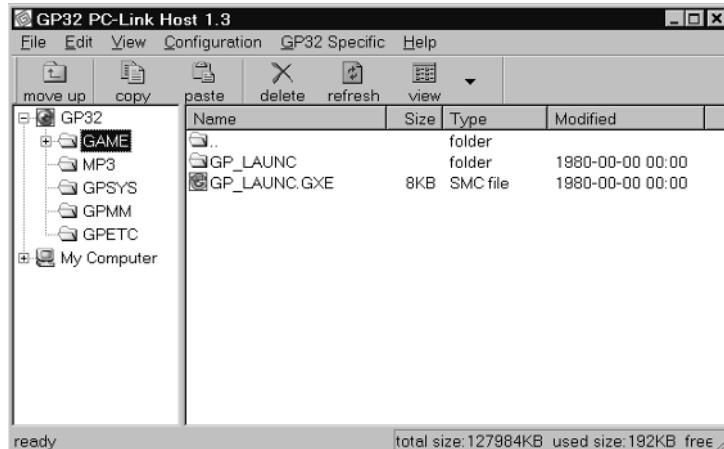
Finally, your GP32 is set up and ready for use! You can load the GP32 with your desired programs and games. Simply use the Install command or drag and drop the desired files into the proper directories on the GP32 device. The files must be loaded into the proper directories on the SMC card or they will not work, so be sure to refer to each particular application's manual or help file. Table 6.1 shows the typical usage of the directories.

Table 6.1 GP32 Directories and Their Typical Functions

Directory	Function
GAME	Commercial games and applications
MP3	MP3 audio files
GPSYS	System files
GPMM	Homebrew (noncommercial) games and applications, FXE files, and emulator ROMs
GPETC	Miscellaneous files

15. Copy the personalized Free Launcher application onto the SMC card. Be sure to maintain the path structure of the files. The gp_launc.gxe file should go into the GAME folder, and gp_launch.gxc should go into the GAME/GP_LAUNCH folder (see Figure 6.9).

Figure 6.9 GP32 PC-Link Host with Free Launcher Installed



16. Close the GP32 link program.
17. Turn off device and remove the USB cable.

Operation of the GP32 is fairly straightforward. Three modes are chosen using the Select and Start buttons. The MP3 mode shows a playlist of all the loaded MP3 files and has a standard user interface for playing the songs (see Figure 6.10). The PC-LINK option, as discussed earlier, waits for a host program on the PC to communicate with the GP32. The GAME mode shows all the loaded commercial games. If the Free Launcher is installed (which it should be if you followed the previous steps), it will also appear (see Figure 6.11). Select the **Free Launcher** and it will list all the noncommercial games loaded on your SMC card.

Enjoy your GP32!

Figure 6.10 GP32 MP3 Mode



Figure 6.11 GP32 Game Mode Showing Free Launcher

NOTE...JOE'S FIRST EXPERIENCE WITH THE GP32



My first encounter with the GP32 was not a good one. A friend of mine had purchased the unit with a singular goal of running an Atari 2600 emulator on it. We spent the night attempting to register the device with Gamepark and trying to figure out how to load games onto the SmartMedia card. The instructions that came with the unit were poorly written and essentially useless. There was an odd bootloader preloaded onto the device; when we turned it on, we didn't get the regular Gamepark screen as we expected (and as is shown in the manual). It turns out that the unit had been modified at the retailer with a Multifirmware loader that is preferred by developers and hobbyists. (See the "Installing the Multifirmware Loader" section in this chapter for more details.)

After we poked around and realized that we could select from three possible boot-up options, things started to come together, but it wasn't pretty. The whole package seemed like a kludge, various programs written by various people. Where was the original software written by Gamepark? Over time, we figured out how to use the PC-Link program to load MP3s and executables onto the GP32. Hearing MP3s being played out of the tiny little speakers of the GP32 was satisfying, but we still weren't able to play any games.

A distribution CD full of GP32 games and programs was included with my friend's order, though he didn't specifically ask for it. We experimented with loading various programs onto the memory card, but even that was confusing, since most games had multiple files that needed to be loaded into specific directories. Suffice it to say, we didn't get many games running.

We finally loaded some Atari 2600 ROMs onto the card and ran GP2600, the Atari 2600 emulator. The only game that actually worked decently for us was Pitfall! An excellent game, but only one of the hundreds that were available for the Atari. The version of GP2600 we used was a public beta, and about 30 games are known to be compatible with the emulator. My friend was disappointed, to say the least, and was dead set on returning his GP32 the next day. I knew I'd be hacking the GP32 for this book, so I convinced him to hold on to it until the chapter was done. Hopefully, after he reads this, he'll be able to play some games!

Opening the GP32 Console

The next three hacks after this one require that you open the GP32 console to gain access to the internal components and circuitry. So, that's what we'll do in this hack. Opening the system is simple, but keeping track of where all the screws go can be confusing.



Preparing for the Hack

The only tool required for this hack is a jeweler's size Phillips screwdriver.

Performing the Hack

WARNING: HARDWARE HARM



The GP32 circuit board contains many electrostatic-sensitive components. You must always be grounded before touching the inside of your GP32 or you risk damaging these components. The easiest way to ground yourself is to buy an anti-static wrist strap and connect it to a grounded source. If you do not have a wrist strap, you can ground yourself by touching a piece of metal (such as the edge of your desk) before touching the GP32's circuitry.

Perform the following:

1. To open the GP32 console, which will give you access to all the internal circuitry, you first need to remove the plastic directional joystick (see Figure 6.12). To do so, carefully pull the joystick straight up and away from the console. The joystick will loosen and finally detach from the actual mechanism that is inside the unit.

Figure 6.12 Remove the Plastic Directional Joystick, Before and After



2. Flip the case upside down, open the battery cover, and remove the two AA batteries (if installed).
3. Remove the six screws holding the case together, denoted in Figure 6.13.

Figure 6.13 Backside of the GP32 Showing Screw Locations



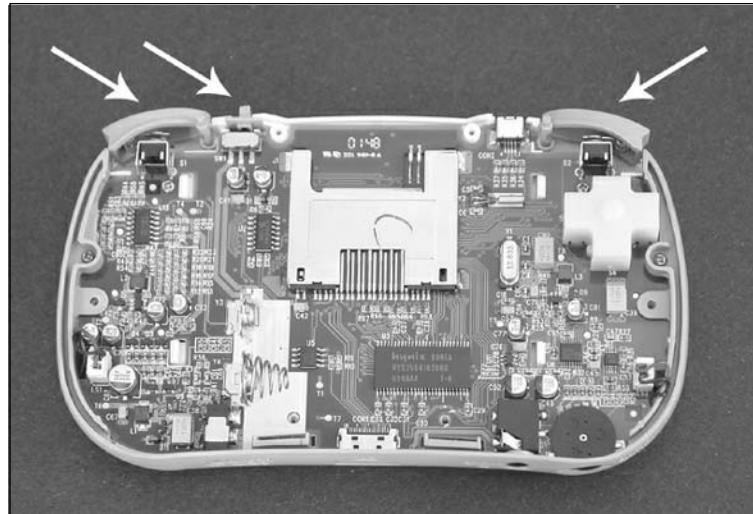
4. The case consists of two plastic pieces, the top and the bottom. After you remove the screws, carefully lift the bottom of the GP32 housing straight upward, separating it from the top (see Figure 6.14).

Figure 6.14 Separate the Bottom and Top Halves of the Console



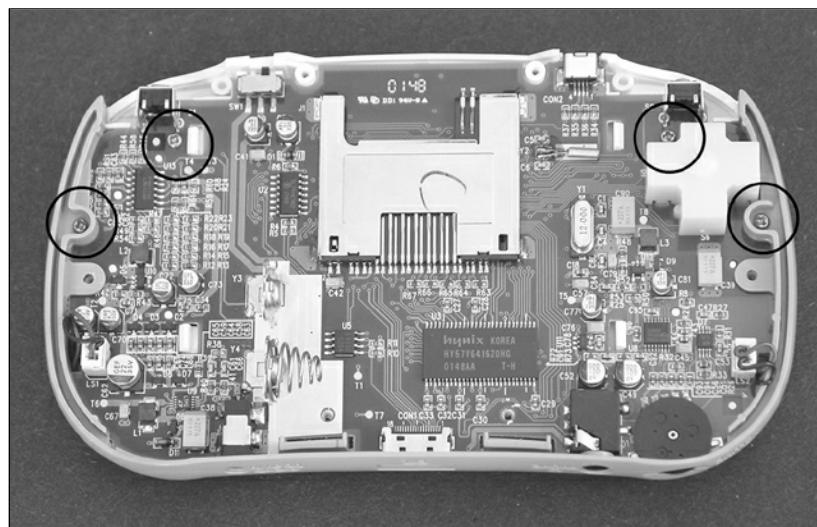
- With the bottom half of the case removed, take off the plastic left and right finger triggers and power switch cover (see Figure 6.15) and set them aside.

Figure 6.15 GP32 Bottom Circuitry with Finger Triggers and Power Switch Cover Denoted



- Next we have to remove the front housing, which will give us access to the LCD screen and front panel buttons. Do so by removing the four screws holding the circuit board to the front housing, denoted in Figure 6.16. These screws will likely be very tight and will take some muscle to loosen.

Figure 6.16 Remove the Four Screws from the Circuit Board



7. Now remove the grey plastic “skirt,” as denoted in Figure 6.17. This skirt, which surrounds the connectors on the back of the GP32, should be pulled straight up and away from the circuit board. Two plastic slots are press-fit into the bottom housing (see Figure 6.18 for a close-up) and you might have to wiggle the skirt gently to loosen it.

Figure 6.17 Remove the Grey Plastic Skirt

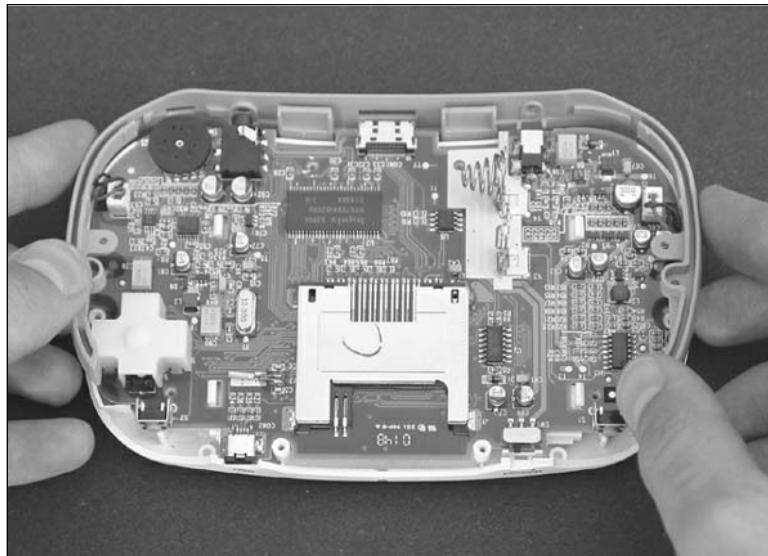
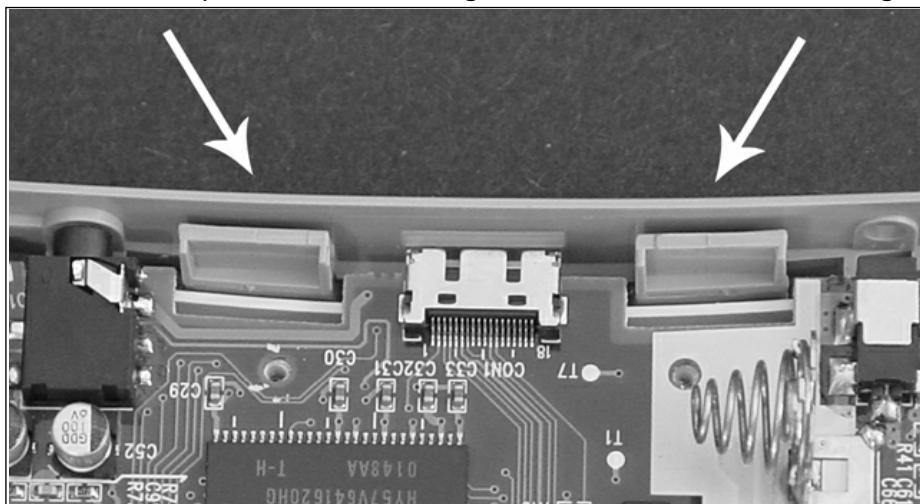


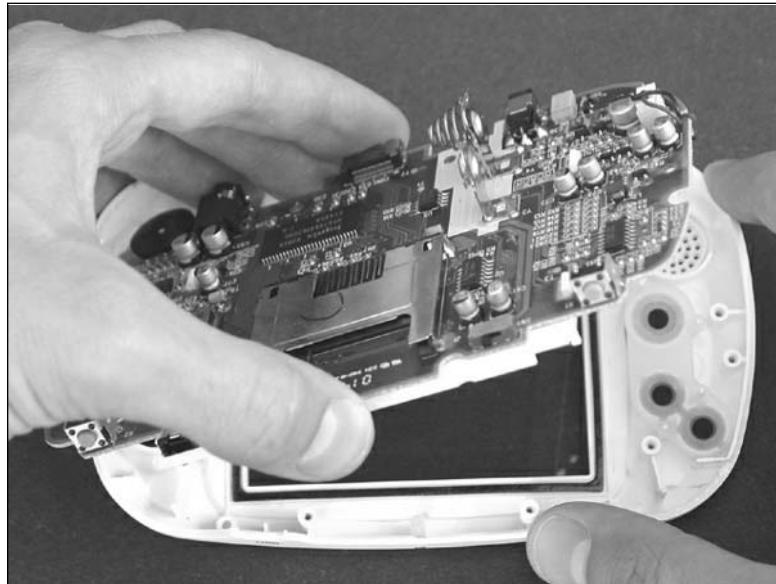
Figure 6.18 Close-Up of the Slots Holding the Skirt to the Bottom Housing



8. With the skirt removed, gently lift the circuitry away from the bottom housing (see Figure 6.19). It might stick a little, so be careful not to flex the circuit board as you are doing so. If

the small speakers on the left and right of the console are glued into place, simply remove the connectors (marked LS1 and LS2 on the circuit board silkscreen) to give you the freedom to remove the circuit board.

Figure 6.19 Remove the Circuit Board from the Bottom Housing



9. From the top housing, if necessary, you can remove the two rubber switch membranes and the plastic A and B buttons from their slots.
10. With the circuit board removed, you should be left with two plastic housing pieces, the plastic skirt, and the complete circuit board (see Figures 6.20, 6.21, and 6.22). This will serve as the starting point for the other hacks and modifications in this chapter.

Figure 6.20 Completely Disassembled GP32



Figure 6.21 GP32 Circuit Board, Front

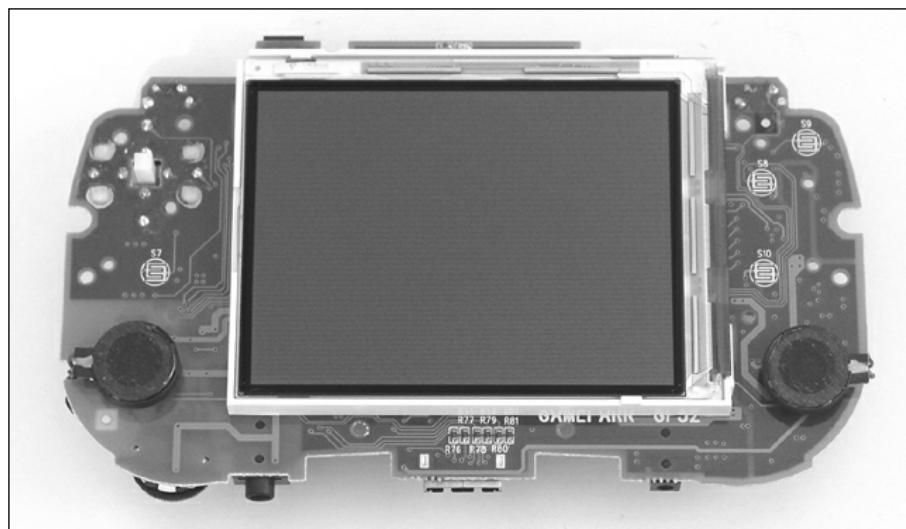
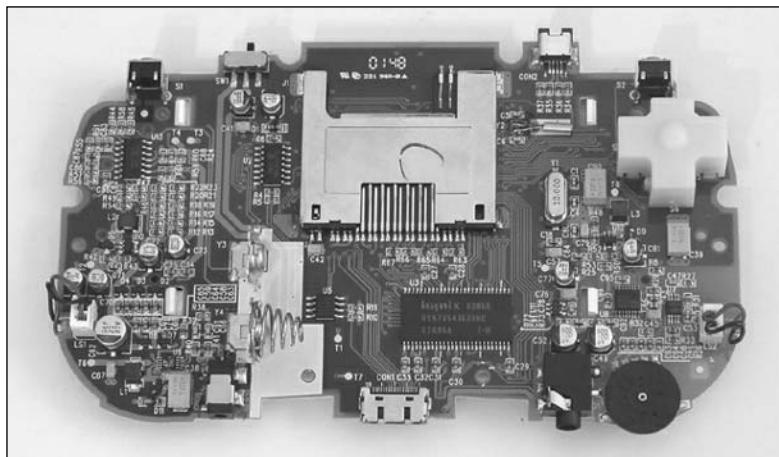


Figure 6.22 GP32 Circuit Board, Back



To put the console back together, simply follow the disassembly steps in reverse order:

1. If applicable, insert the plastic A and B buttons back into their respective slots. (As you're looking at the front of the GP32 console, the A button should be in the upper right and the B button should be in the lower left.) Reattach the rubber switch membranes.
2. Place the circuit board into the front housing, taking care not to crush the speaker wires.
3. Reattach the plastic skirt and screw the four screws back into the circuit board and plastic skirt.
4. Replace the power switch cover and left and right finger triggers, then attach the bottom half of the GP32 housing and secure it into place with the six screws.
5. Finally, replace the plastic directional joystick.

Your GP32 should now be reassembled!

Replacing the GP32 Screen Cover

Due to the portable nature of the GP32, it is only a matter of time before your screen will become scratched or dirty enough to affect game play and the enjoyment of your console. This simple hack will guide you through the process of replacing the plastic screen cover of the GP32 that is used to protect the LCD display. If done correctly, replacing the screen cover will help make your display appear as it did when you first bought your GP32.



Preparing for the Hack

The only part required for this hack is a replacement GP32 screen cover (manufactured by Gamepark and available from www.gbax.com, as shown in Figure 6.23). The only tool you'll need is a jeweler's size Phillips screwdriver to open the case.

Figure 6.23 GP32 Replacement Screen Cover



Performing the Hack

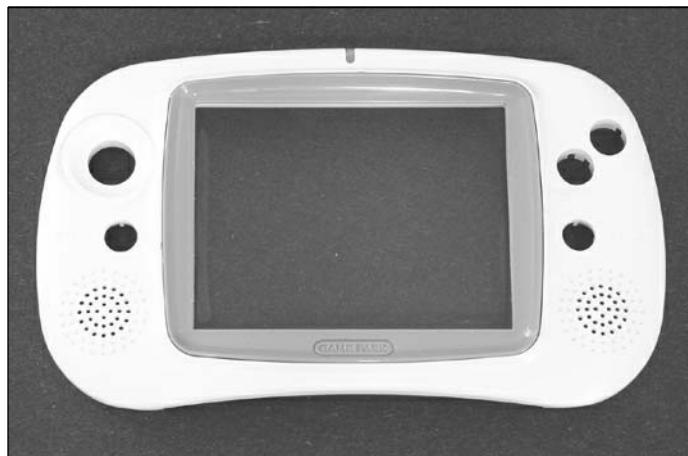
Perform the following:

WARNING: HARDWARE HARM



When you are performing the GP32 screen cover replacement hack, be sure that your workspace is clean and completely free of dust and dirt. Any small specs of debris that come in contact with the new screen or the LCD panel could cause scratching and undesired display artifacts.

1. Open your GP32 console as described in the “Opening the GP32 Console” section earlier in this chapter. We’ll be working with the front half of the console housing that contains the plastic protective screen cover (see Figure 6.24).

Figure 6.24 GP32 Top Housing with Screen Cover

2. Carefully push the screen cover away from the unit, using your fingers at the inside corners of the GP32 top housing (see Figures 6.25 and 6.26). The screen cover and gray plastic LCD bezel are attached to the front of the GP32 with strong adhesive tape, so take care not to bend or warp any of the plastic while you are attempting to free it.

Figure 6.25 Pushing the Plastic Bezel and Screen Cover Away from the Housing

Figure 6.26 Removing the Plastic Bezel and Screen Cover



3. Remove the screen cover from the plastic bezel (see Figure 6.27) and discard it.
4. Remove the protective film from both sides of the new GP32 screen cover (see Figure 6.28) and place it into the bezel where the old screen cover was located. Once the protective film is removed from the new screen cover, be very careful not to scratch the screen or get any dirt, dust, or fingerprints on it, since doing so will affect the visual quality of the GP32 once you put everything back together.

Figure 6.27 Removing the Old Screen Cover from the Plastic Bezel



Figure 6.28 Peeling the Protective Film from the New Screen Cover

5. With the new screen cover in place in the bezel, reattach the bezel to the front of the plastic housing and secure it to the already existing adhesive tape by pressing firmly around all four edges, making sure to push in the four corners as well (see Figure 6.29 and 6.30). If the adhesive tape used to keep the bezel in place is no longer sticking, you can use a small dab of Super Glue or a length of double-sided Scotch tape to reaffix it.

Figure 6.29 Placing the Bezel and New Screen Cover onto the GP32, Step 1

Figure 6.30 Placing the Bezel and New Screen Cover onto the GP32, Step 2

6. You can now reassemble your GP32 as described in the previous “Opening the GP32 Console” section. Before reassembly, you might want to use a can of compressed air to clean off any dust that might have been attracted to the new screen.

The hack is complete and you can now enjoy your new, crystal-clear GP32 screen!

Repairing Your Buttons

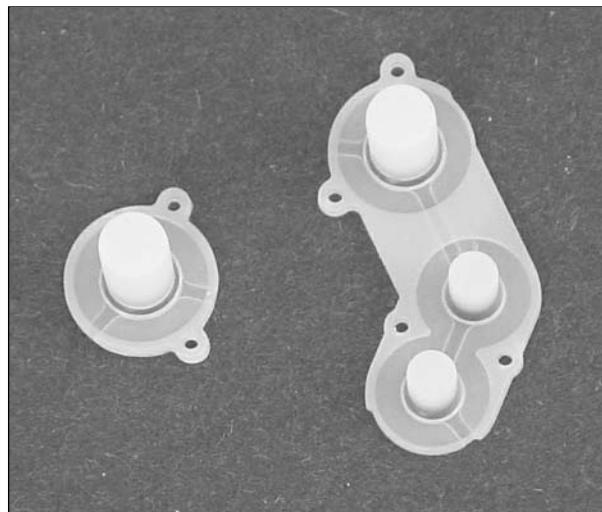
With heavy use of the GP32, the contacts between the button membrane and the GP32 circuit board can become worn. This prevents a good electrical connection when the buttons are pressed, which can result in intermittent problems when you’re using the GP32 buttons. In addition, there are rumors of production flaws in some of the GP32 units due to the small-quantity manufacturing runs. One such flaw includes nonresponsive buttons.

This extremely simple hack will guide you through the process of replacing the button membranes to make your buttons react as well as they did when they were new.



Preparing for the Hack

The only parts required for this hack are the replacement button membranes for the GP32 (manufactured by Gamepark and available from www.gbax.com; see Figure 6.31). The only tool you’ll need is a jeweler’s size Phillips screwdriver to open the case.

Figure 6.31 GP32 Replacement Button Membranes

Performing the Hack

Perform the following:

1. Open your GP32 console as described in the “Opening the GP32 Console” section earlier in this chapter. With the circuit board removed, you should be left just with the GP32 top plastic housing, as shown in Figure 6.32.
2. Remove the two button membranes, as denoted in Figure 6.32, by lifting them directly up and away from the housing. Each membrane is held in place by plastic posts. Take care to not remove or lose the plastic A and B buttons that are underneath the larger button membrane, shown in the lower-right corner of Figure 6.32. When the old button membranes have been removed, your housing should resemble the one shown in Figure 6.33.

Figure 6.32 GP32 Top Housing Showing Button Membranes

Figure 6.33 GP32 Top Housing with Old Button Membranes Removed

3. The final step in the hack is to simply place the new button membranes in the locations of the old ones. When inserting the new button membranes, be careful not to tear the rubber as you are pushing them onto the plastic posts. Make sure they are held firmly in place by the posts. Your housing will once again resemble the one shown in Figure 6.32.
4. You can now reassemble your GP32 as described in the previous “Opening the GP32 Console” section.

Enjoy your new, responsive buttons!

Accelerating Your GP32 (CPU Core Voltage Increase)

The GP32 hardware is based on a Samsung S3C2400X01 (ARM920T core) 32-bit processor running at a software-selectable setting between 20 MHz and 133 MHz. By default, the GP32 runs at 66 MHz. All GP32 units can be overclocked to 133 MHz with no problems, but some applications (particularly emulators and media players) require a higher clock speed to maintain accuracy and functionality. Without modification, many GP32 consoles that are overclocked above 133 MHz tend to exhibit erratic behavior.

This hack will increase the voltage of the CPU core and will allow your GP32 to overclock reliably to 166 MHz (and possibly up to 180 MHz!). The hack is based on Cobbleware’s “Overclocking by Changing the CPU Voltage” page at www.cobbleware.com/gp32/gp32oc.html.

WARNING: HARDWARE HARM

Overclocking your GP32 console can potentially damage the circuitry (due to overheating and voltages being above the recommended operating levels). It is recommended that you do not clock your system above 133 MHz unless it's required by a particular game or application.



Preparing for the Hack

For this hack, you only need two components:

- A 100k ohm, 1%, size 0603, 1/16W resistor (Digi-Key part #P100KHCT-ND)
- A 56k ohm, 5%, size 0603, 1/16W resistor (Digi-Key part #P56KGCT-ND)

The tools required for the hack are:

- Jeweler's size Phillips screwdriver
- Soldering iron and solder
- Tweezers

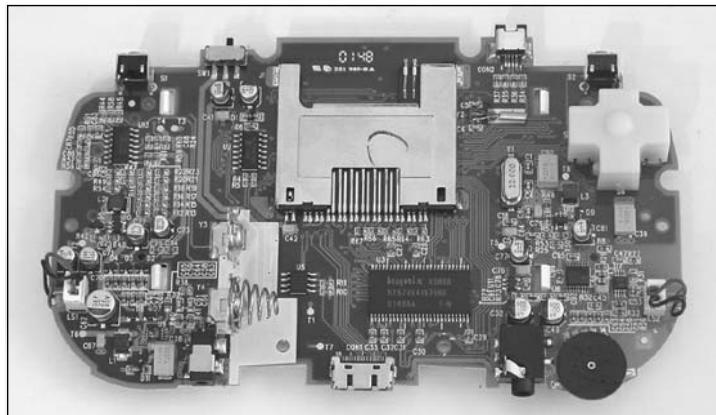
This hack consists of simply replacing two resistors on the circuit board. However, this task can be tricky because the parts are extremely small (1.6mm long by 0.8mm wide)!

Performing the Hack

Perform the following:

1. Open your GP32 console as described in the “Opening the GP32 Console” section earlier in this chapter. With the bottom housing removed, you should have complete access to the GP32 circuit board, as shown in Figure 6.34.

Figure 6.34 GP32 Circuit Board, Back



2. We need to remove two resistors from the GP32 circuit board and replace them with new ones. Figure 6.35 shows a close-up of the board with the two resistors to remove.

Unsolder the two connections of each resistor, carefully pulling the resistors off the board, and discard them. It might help to use the tweezers to hold onto the part as you are desoldering it (see Figure 6.36). The resistors are denoted by R48 and R52 on the silkscreen. With the resistors removed, your board should resemble the one shown in Figure 6.37.

WARNING: HARDWARE HARM



Be careful when removing the resistors from the board, because you don't want to pull up the solder pads from the circuit board. Try and remove as much solder as possible from the joints, then carefully pull the resistors up from the board while heating the connections.

Figure 6.35 GP32 Circuit Board Showing the Two Resistors to Remove

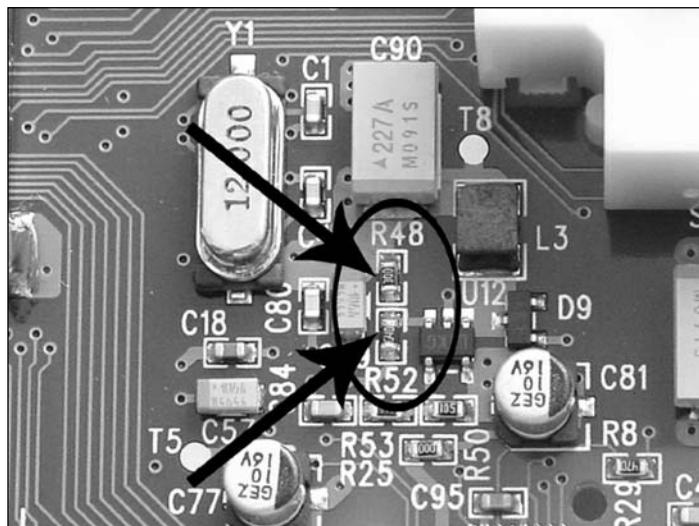
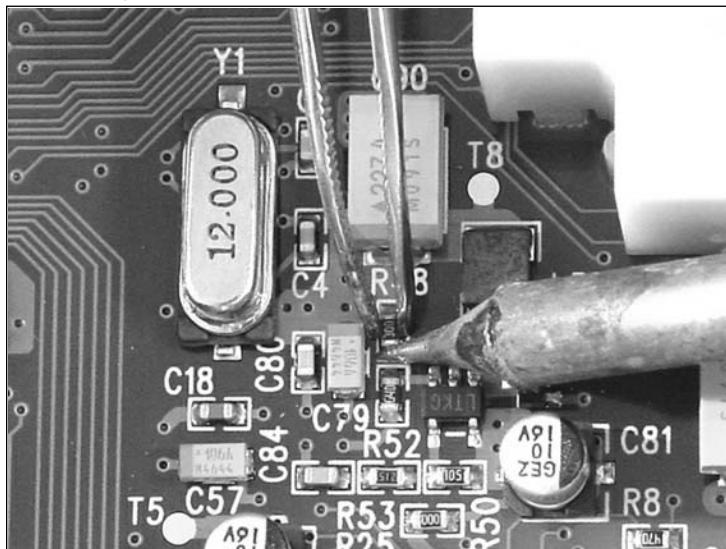
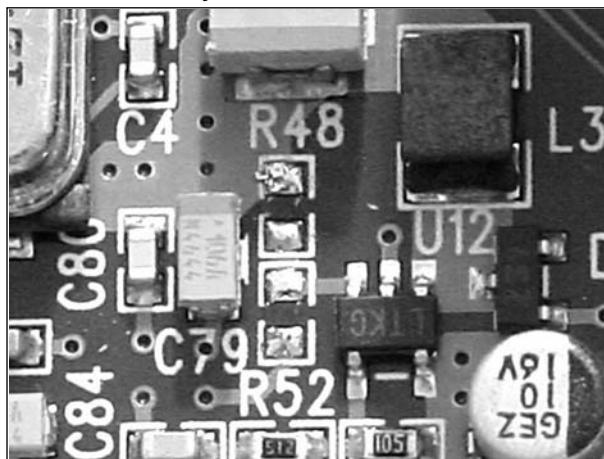
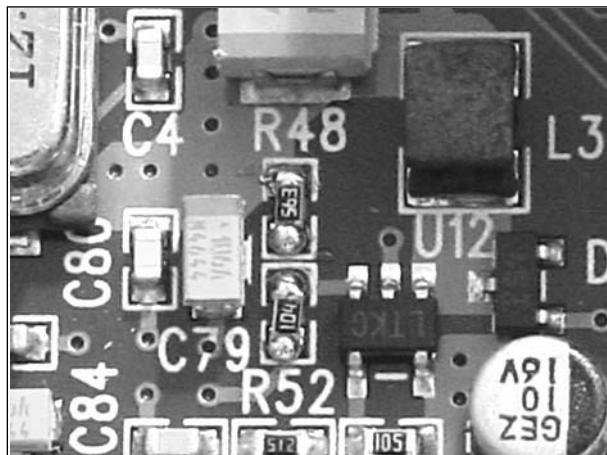


Figure 6.36 Removing the Resistors**Figure 6.37** Resistors Successfully Removed

3. Next, put the new resistors into the location of the old ones and solder them into place. The new resistors can be placed in either orientation—the polarity does not matter. The new 56k resistor should be soldered into R48's location (the top resistor, as shown in Figure 6.35), and the 100k resistor should be soldered into R52's location (the bottom resistor, as shown in Figure 6.35). When complete, your modified board should resemble the one shown in Figure 6.38.

Figure 6.38 New Resistors Soldered into Place

4. You can now reassemble your GP32 as described in the previous “Opening the GP32 Console” section.

Your GP32 will now have a higher CPU core voltage and should give you a more reliable result when you overclock your console above 133 MHz!

Under the Hood: How the Hack Works

The GP32’s clock speed is set by each individual application as it runs. Some programmers choose to release multiple versions of an application, one for each clock speed (for example, 133 MHz, 150 MHz, and 166 MHz). Other programs come with a slider bar, which allows users to set the speed themselves (such as with GP Cinema, a freeware DivX movie player available from <http://joygp.entreware.com>, which allows a software-selectable clock speed of 100 MHz, 120 MHz, 133 MHz, 150 MHz, or 166 MHz).

Although all GP32 units can be overclocked to 133 MHz without this hack, overclocking to higher frequencies typically results in erratic behavior of the device. The premise of this hack is that the expected maximum clock speed will increase as the CPU core voltage increases. This type of hack has been done for years in the PC overclocking community, so it is a relatively simple task to apply the theory to the GP32.

Samsung’s S3C2400x Processor User’s Manual (available from www.cs.helsinki.fi/u/jikorhon/gp32/dl/um_s3c2400x_rev1.1.pdf) states that V_{DDi} , the core CPU voltage input, has a recommended operating range between 1.65V and 1.95V. A core voltage of 1.8V is ideal, though its maximum rating is 2.7V (after which the CPU could be irreversibly damaged). With this hack, we aim to increase V_{DDi} to the high edge of its recommended operating range, 1.95V.

In the GP32, a Linear Technology LTC1701 1 MHz Step-Down DC-DC Converter (denoted with LTKG on the top of its SOT-23 package, clearly seen in Figure 6.38) is used to regulate the core voltage. A DC-DC converter is essentially a voltage regulator that takes an input voltage on one pin

and regulates it to a fixed voltage that is output on another pin. With the LTC1701, the output voltage is adjustable and is set by a resistor divider configuration according to the following formula:

$$V_{OUT} = 1.25V * (1 + (R2 / R1))$$

where

$R1$ = Resistor 1 in ohms = $R52$ in the GP32

$R2$ = Resistor 2 in ohms = $R48$ in the GP32

In the GP32's original configuration, $R48$ is 200k and $R52$ is 470k ohm. Plugging those values into the equation gives us:

$$V_{OUT} = 1.25V * (1 + (200000 / 470000)) = 1.78V$$

So, in the stock configuration, V_{DDi} to the CPU is 1.78V, almost exactly the ideal voltage of 1.8V recommended by the S3C2400x data sheet.

Replacing $R48$ and $R52$ with 56k and 100k ohm, respectively, as we do in this hack, will result in an output voltage of:

$$V_{OUT} = 1.25V * (1 + (56000 / 100000)) = 1.95V$$

Since we are changing the voltage regulator output to provide a V_{DDi} of 1.95V, we are right at the edge of the CPU's recommended operating range, just where we wanted to be! Based on component tolerances, the actual output voltage could be slightly higher or lower, but we are well within the absolute maximum rating of the CPU. It is unlikely that any long-term damage of the CPU will occur as a result of this hack.

Creating a DC Power Adapter

Unless you live in Korea, it is difficult to obtain an official GP32 DC power adapter that will allow you to power the GP32 from a wall outlet instead of using batteries. This hack will guide you through the process of creating your own GP32 DC power adapter from a standard Nintendo Game Boy-compatible one.

Preparing for the Hack

For this hack, the only component you will need is a Nintendo Game Boy-compatible DC power supply adapter. We're using a MadCatz GBA Power Solution kit (which includes the necessary 100mA DC power adapter, shown in Figures 6.39 and 6.40), but you could use a variety of other third-party adapters, including:

- Radio Shack 3V, 350mA Power Adapter (for use with Nintendo Color and Pocket Game Boy), part #273-1750, \$9.99

- Radio Shack 3V, 500mA Power Adapter (part #273-1755, \$15.99) with Adaptaplug A (part #273-1704, \$4.99)
- Pelican Accessories AC/DC Power Pak

Other third-party and older MadCatz AC adapters will also work properly. The most important criteria for the DC power adapter are that it provides 3V DC at 100mA or greater and that the connector tip has a 2.3mm outer diameter (OD) and 0.7mm inner diameter (ID). Although the back of the GP32 states that it requires a 3V DC input with a 300mA current capacity, a 100mA supply appears to work just fine.

Figure 6.39 MadCatz GBA Power Solution (Including 100mA DC Power Adapter)



Figure 6.40 MadCatz GBA DC Power Adapter, Close-Up



The tools required for this hack are:

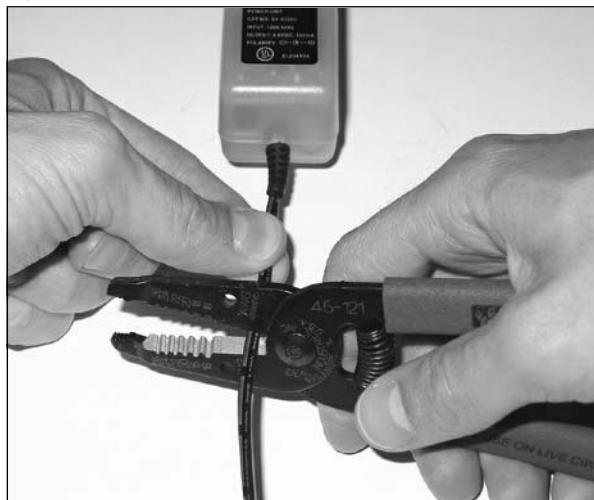
- X-ACTO knife or hobby blade
- Wire cutter/stripper
- Soldering iron
- Heat gun
- Heat-shrink tubing (two lengths of small-diameter, one length of large-diameter)

Performing the Hack

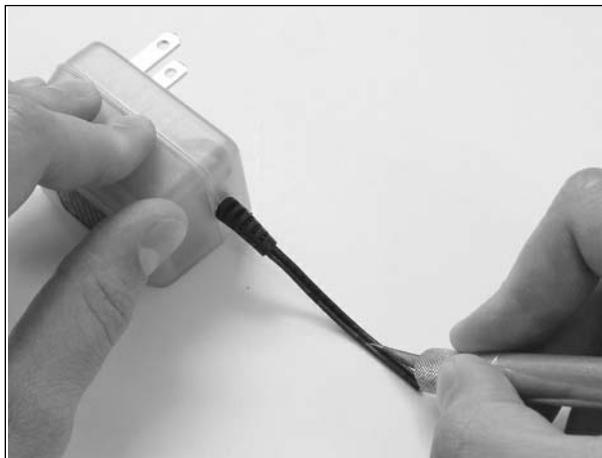
Perform the following:

1. Cut the cable and connector off the power adapter. As shown in Figure 6.41, cut the wire about 2 inches away from the actual power adapter.

Figure 6.41 Cutting the Cable Off the Power Adapter



2. Using a hobby blade, split the two plastic-coated leads on both pieces and pull them apart into 1-inch lengths (see Figure 6.42).

Figure 6.42 Splitting the Cable

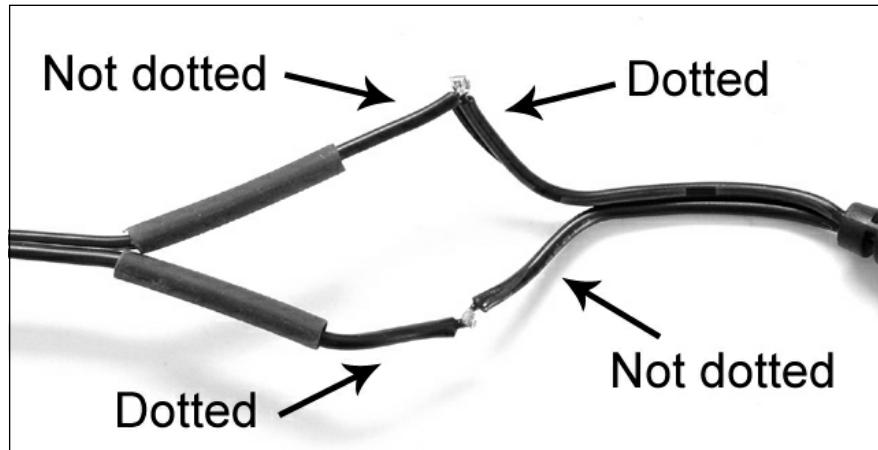
3. Strip 1/4 inch of the plastic to expose the wire beneath. Twist the braided wires together so they don't fray. Insert the two small pieces of shrink tubing onto the wires on the adapter side, and insert the larger piece of shrink tubing onto the other side, pushing it down the wire and out of the way. Your current setup should resemble the one shown in Figure 6.43.

Figure 6.43 Successful Preparation of the Wires

4. Twist together the wire marked with the dotted line on the power supply to the wire *without* the dotted line on the power plug.
5. Twist the two other remaining wires together.

6. Solder both connections, taking care to keep the shrink tubing out of the way and not to apply too much heat, which could melt the plastic insulation on the wires.
7. Clip the soldered connection down to remove any unsoldered wire (see Figure 6.44).

Figure 6.44 Wires Soldered Together



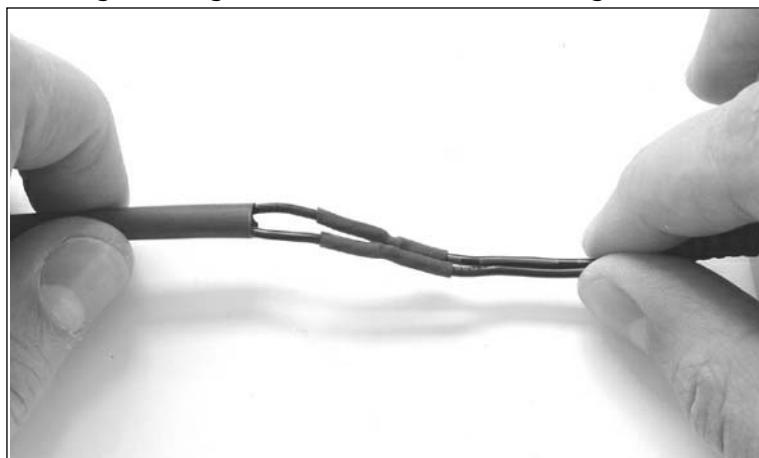
8. To complete the hack, we need to shrink the heat-shrink tubing around our new connections, to prevent any electrical shorts. This step could also be achieved using electrical tape, though it is not as visually pleasing and the tape can also loosen, fall off, or dry out over time, leading to failures later on. Slide the two small pieces of heat-shrink over the newly soldered connections.
9. Next, use the heat gun and apply an even heat to all sides of the tubing, until they shrink completely around the connections (see Figure 6.45).

Figure 6.45 Applying Heat to the Heat-Shrink Tubing



10. Once the tubing has cooled, slide the larger piece of tubing over both of the wires (see Figure 6.46). Hopefully, your piece of large heat-shrink tubing is long enough to cover the two connections and any remaining space in the wire that you had cut. Use the heat gun again to shrink the tubing.

Figure 6.46 Sliding the Larger Piece of Heat-Shrink Tubing Over the Wires



11. When the large piece of heat shrink has successfully been shrunk into place, the hack is complete (see Figure 6.47)!

Figure 6.47 Completed Hack: MadCatz GBA DC Power Adapter with Reversed



Under the Hood: How the Hack Works

This simple hack works by reversing the polarity of a Nintendo Game Boy-compatible DC power supply adapter to function with the GP32.

The GP32's power adapter input (see Figure 6.48) shows the industry-standard symbol for a connector's polarity. In this case, the negative (ground) lead is on the tip and the positive lead is on the shield. Typically, most consumer electronic devices around the world, including the Nintendo Game Boy, place the positive lead on the tip and the negative (ground) lead on the shield. So, this hack just reverses the polarity of the DC power supply adapter to allow the positive lead to be on the outside of the plug and the negative (ground) lead to be on the inside, as the GP32 expects.

Figure 6.48 GP32 Power Adapter Input, Close-Up



Installing the Multifirmware Loader

This hack will guide you through the process of installing Mr. Spiv's custom Multifirmware loader onto your GP32, replacing the standard Gamepark firmware. Although your system will run fine without the Multifirmware hack, this modified firmware makes game development much easier because it allows you to choose from alternative front ends when the console is first turned on. The version of Multifirmware we're using for this hack provides a boot selector for the following front ends:

- Gamepark's factory-original GP32 firmware (version 1.57e)
- Pacrom v0.31b, a Windows Explorer-styled file manager used to easily select and run programs
- Wind-ups v0.9+, a GUI desktop resembling the beloved Windows operating system (www.wind-ups.net)
- PC-Link v1.01, an easy way to enter the GP32's PC LINK mode to transfer data to and from the GP32

Multifirmware also contains GNU Project Debugging (GDB) support with Mithris' GDB-Stub (<http://gp32.misantrop.org/Down.php>).



Preparing for the Hack

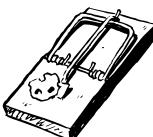
For this hack, you will need your GP32 and the following two files:

- **Dark Fader's DumpFW** firmware backup tool, available from <http://darkfader.net/gp32/files/DumpFW.fxe.zip>.
- **Mr. Spiv's Multifirmware (MultiFW2)**, available from www.cs.helsinki.fi/u/jikorhon/condev/gp32/dl/multifw2.fxe. The MultiFW2 tool makes firmware upgrading very simple. It contains the firmware reprogramming code, verification routines, and the custom firmware image, all in one single GP32 executable package. Other Multifirmware versions are available, but the MultiFW2 seems to be stable and used by most GP32 hobbyists.

Performing the Hack

This hack is split into two sections. The first is to back up the original GP32 firmware in case disaster strikes during the upgrade process. The second is to actually reprogram the new firmware into the GP32 device.

WARNING: HARDWARE HARM



Incorrect reprogramming of the GP32 firmware could cause irreparable damage to the console. Proceed with caution!

Backing Up Your Firmware

The first thing to do is ensure that you have a backup of your GP32's firmware. In case something goes wrong during the reprogramming process, you might be able to recover your system by reloading the original firmware. We will be using the DumpFW tool, which will write the firmware to the SMC card installed in your GP32. Be sure that your SMC card has 512 KB of free space before running the tool.

Perform the following:

1. Extract the contents of the DumpFW.fxe.zip file into a temporary directory (there should be one file, DumpFW.fxe).
2. Using the PC-Link Host software on your PC (or other desired operating system), install the DumpFW.fxe tool into the GPMM folder of your SMC. Exit PC-Link Host when the transfer is completed.

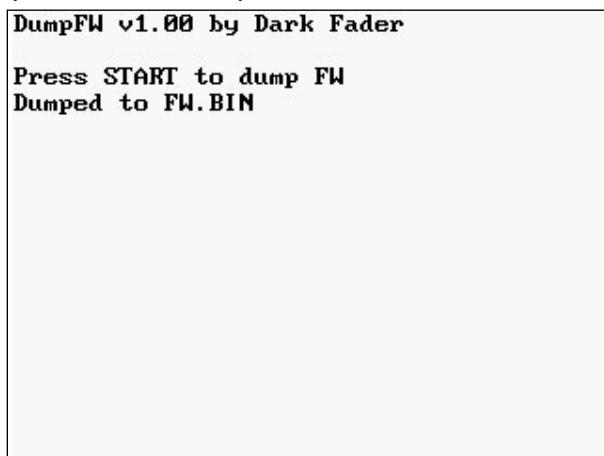
3. Using Free Loader on your GP32, run the Dump FirmWare application. You should be prompted with the start-up screen (see Figure 6.49).

Figure 6.49 DumpFW Start-Up Screen



4. Press the **Start** button on your GP32 to begin the firmware backup process. It should take a few seconds to copy the firmware from your GP32 to the SMC card. When it's done, a notification message will appear on the screen (see Figure 6.50).

Figure 6.50 DumpFW Process Completed



5. The firmware backup, FW.BIN, has been stored in the root folder of the SMC card. Using the PC-Link Host software, copy the file from the GP32 to your PC and store it in a safe place.

Reprogramming (Flashing) the New Firmware

Now you're ready to install Multifirmware! Before beginning, ensure that you have a new set of batteries installed in the console or use a DC power adapter, which will allow you to power the GP32 from a wall outlet instead of using batteries (see the previous "Creating a DC Power Adapter" hack to create your own). If the reprogramming process is interrupted, your system could be left in an unknown state with no method of recovering it!

Perform the following:

1. Using the PC-Link Host software on your PC (or other desired operating system), install the **multifw2.fxe** tool into the GPMM folder of your SMC. Exit PC-Link Host when the transfer is completed.
2. Using Free Loader on your GP32, run the **MultiFw2 Flash Tool** application. As soon as you run the program, the upgrade process will begin. Do not turn off your GP32 until the process is completed! You will see the status of each step, from erasing the old firmware to programming and verifying the new firmware (see Figures 6.51 and 6.52). When it's done, a notification message will appear on the screen (see Figure 6.53).

Figure 6.51 Multifirmware Upgrade in Progress, Deleting the Existing Firmware

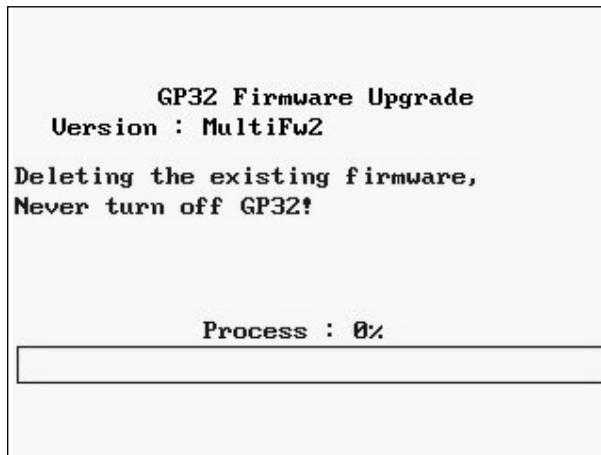


Figure 6.52 Multifirmware Upgrade in Progress, Programming the New Firmware

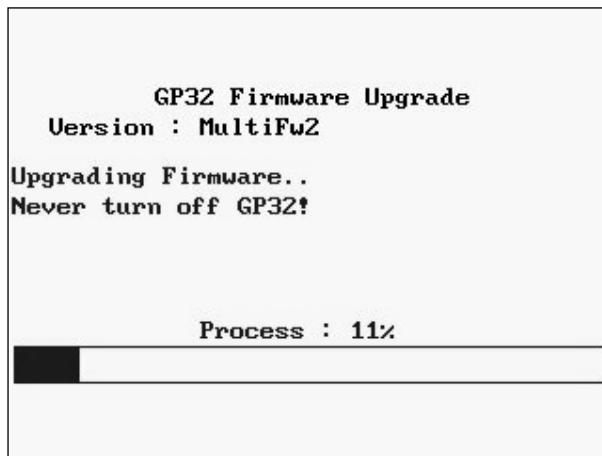
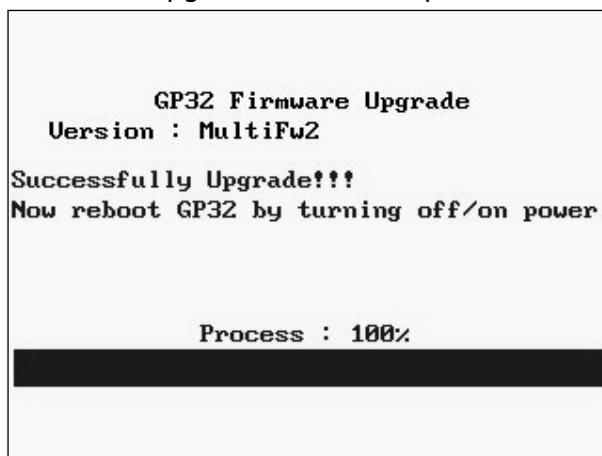
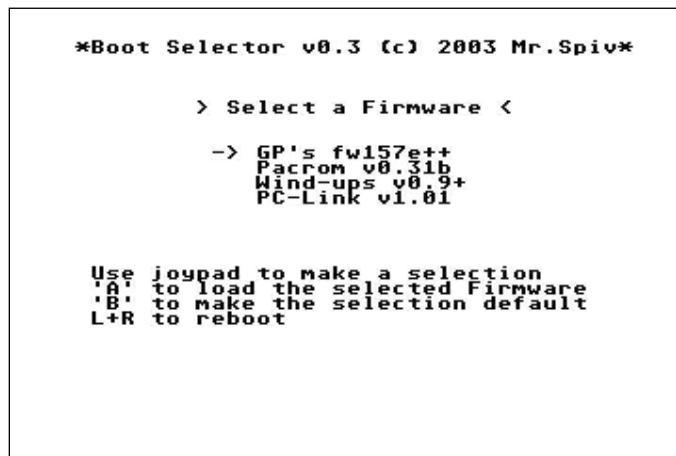


Figure 6.53 Multifirmware Upgrade Process Completed



3. If everything worked properly, you should have the Multifirmware loaded onto your GP32. To make sure, turn off the unit and turn it back on. You should be prompted with the boot selector menu (see Figure 6.54). You can now enjoy your new Multifirmware!

Figure 6.54 The MultiFW2 Boot Selector Menu

As the text at the bottom of the screen states, use the joypad to select a front end option and press the **A** button to load it. Pressing the **B** button will configure the GP32 to boot straight to that selection upon power-up. Your boot selector settings are stored in the GP32’s internal EEPROM, so your default application selection will be saved until you change it again from the boot selector menu. (To get back to the menu after a selection has been made, hold the **Select** button while turning on the GP32.) This is extremely useful as it removes the need to go through the repetitive process of choosing a specific front end each time the console is turned on.

If the upgrade didn’t work but you *can* still boot your GP32, you can try to run through the steps of this hack again and cross your fingers. Or you can reprogram your original firmware back into the device using the GP32 Firmware Upgrade Utility, available at <http://darkfader.net/gp32/files/fw.fxe.zip>.

If the upgrade didn’t work and you *can’t* boot your GP32, chances are that your firmware is corrupted. This is a bad sign. You can attempt to use Cobbleware’s GP32 JTAG hack (see the “Other Hacks” section of this chapter) to reload the firmware, or you can try to obtain some help from the resources listed in the “Homebrew Game Development” section of this chapter.

Homebrew Game Development

Compared to many current game console vendors who object to experimenting with and writing games for their machines, the GP32 is an open platform with an open-minded and supportive development community.

Most of the GP32’s homebrew development community seems to be focused on creating emulators for the system, so if you’re a gamer hoping to have a portable emulation console, you’re in luck. Videogame emulators exist for arcade games (using MAME, the Multiple Arcade Machine Emulator) and just about every game system, including the Atari 2600, Atari 5200, NES, Super NES, Nintendo Game Boy and Game Boy Advance, Sega Genesis, NEC TurboGrafx-16, NEC PC Engine,

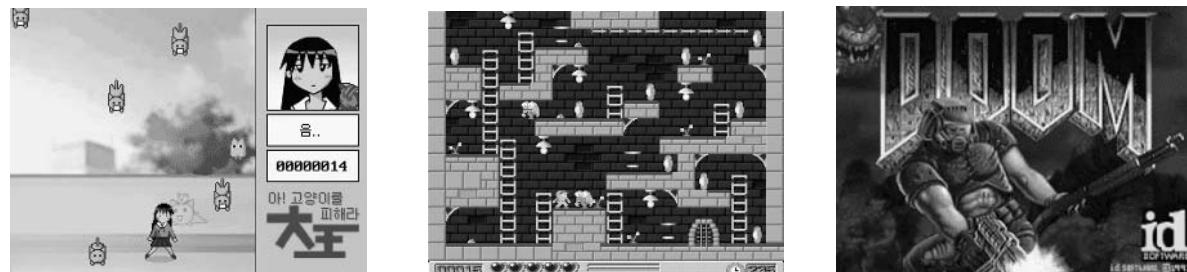
Commodore C64, Atari ST, and Sinclair ZX Spectrum, to name a few (see Figure 6.55). There is also a Script Creation Utility for Maniac Mansion (SCUMM) “virtual machine,” which allows the GP32 to run all of the classic LucasArts adventure games, including Monkey Island, Indiana Jones, Day of the Tentacle, and The Dig. The quality, accuracy, and progress of each emulator may vary, since they are hobbyist projects released for free to fellow gamers, but in general these games are truly amazing.

Figure 6.55 GP32 Homebrew Emulators, Left: Super Mario Bros. on Little John (NES), Right: Pitfall! on GP2600 (Atari 2600)



Many other homebrew developers are focused on writing custom games for the GP32 or demos that show off the programmer’s skill and take the capabilities of the GP32 hardware to its limit. Even though hobbyists write these games in their spare time, many of the games rival the quality of those that are commercially available (see Figure 6.56). Hundreds of homebrew applications have been created and are available for free; see the resources at the end of this section.

Figure 6.56 GP32 Homebrew Games, Left: Ah! Catsmanga Daioh, Center: Crazy Jack, Right: DOOM



The development toolchain for the GP32 typically consists of two core tools, which can be installed on Windows- or Linux-based platforms:

- **DevKitAdvance** A GCC (GNU C Compiler)-based Software Development Kit for the Nintendo Game Boy Advance, available from <http://devkitadv.sourceforge.net>

- **Christian Nowak's GP32 add-on package** Turns DevKitAdvance into a GP32 compiler, available from <http://chn.roarvgm.com/#gp32>.

The GP32 uses a number of file extensions to identify files. Table 6.2 provides a list of the GP32/PC-related file extensions that you may encounter during development and experimentation with the console.

Table 6.2 GP32 and PC-Related File Extensions (from <http://darkfader.net/gp32>)

File Extension	Description
GXB, AXF, BIN	ARM eXecutable Format, a plain binary executable for GP32.
GXE	Gamepark eXecutable Encryption, an information file containing username, icon, and encryption data.
GXC	Gamepark eXecutable Contents, contains the remaining part of the executable data.
GXF	Gamepark eXecutable Free/Format, a plain executable with icon/application information.
FXE	Free eXecutable Encrypted, these files, typically homebrew software, are encrypted for use with GP32's Free Launcher tool.
GPS	Gamepark uSer, contains the username, password, and GP32 ID of the console.
GPM	Gamepark Media, contains the name of the game, size, and SMC ID.
ZPK	ZPacKed archive, an archive containing an encrypted application.
FPK	Free zPacKed archive, an archive containing a free application.
GPK	Unencrypted zPacK archive.
SMC, GP	SmartMedia Card memory dump file.
ELF	Link output format that typically contains debugging information.
C, CPP	C/C++ source code file.
H	Source code header file (usually included with a C/C++ project).
O, OBJ	Object file (compiled source in a binary format).
S, ASM	Assembly-language source code file.

If you'd like to learn more about creating homebrew games for the GP32, the following links should help you out:

- **Mr. Spiv's GP32 Development Page, www.cs.helsinki.fi/u/jikorhon/condev/gp32** Arguably the best source of technical GP32-related information on the Web. Contains bundles of news, hardware specifications, development tools, homebrew programs, example source code, and much more. A must-see for any aspiring GP32 hardware hacker or homebrew game developer.

- **GP32 Devr's, www.devrs.com/gp32** News and information related to GP32 hardware and software development, featuring an exhaustive development-related Frequently Asked Questions (FAQ) section.
- **Dark Fader's Gamepark 32, <http://darkfader.net/gp32>** Good site containing a wealth of technical information for the GP32. The site hasn't been updated in a while, but a lot of the technical details are still relevant and worth a read.
- **Building a GP32 GCC Toolchain in Linux (for GP32 application development), www.cobbleware.com/gp32/gp32toolchain.html** Technical information on configuring a development toolchain for building GP32 applications on a Linux platform.
- **gp32dev Yahoo! Group, <http://groups.yahoo.com/group/gp32dev>** A public forum related to the discussion of GP32 homebrew game development.
- **#gp32dev IRC (Internet Relay Chat) Channel** Located on the EFnet server (<irc.efnet.org>), the #gp32dev channel is dedicated to discussing programming and development for the GP32 console.
- **GP32Emu, www.gp32emu.com** Comprehensive site for all things related to emulation and homebrew game development for the GP32. Featuring news, forums, links, and GP32 games and applications.
- **GeePee32 Emulator, <http://users.skynet.be/firefly/gp32>** The first and currently only available emulator for Windows and Linux platforms. Written by Firefly. To function properly, this emulator requires the original GP32 firmware image, which can be obtained from <http://darkfader.net/gp32/files/fw157e.zip> or dumped from your own device using Dark Fader's DumpFW tool, available at <http://darkfader.net/gp32/files/DumpFW.fxe.zip>.
- **retrodev.info, www.retrodev.info** Home of many popular console emulators, all written for the GP32, including fGEN32 (Sega Genesis/Megadrive), fCOL32 (ColecoVision), fSMS32 (Sega Master System), fGB32 (Nintendo Game Boy) and fNES32 (Nintendo NES).
- **GPAdvance, <http://gpadvance.sourceforge.net>** A Nintendo Game Boy Advance emulator for the GP32. Development is ongoing for this adventurous project.
- **PDRoms, www.pdroms.de/typ.php?system=GP32** Web site devoted to the collection of legal homebrew games for just about every videogame console in existence. The GP32 section has over 250 links to applications, emulators, demos, and games.
- **gpQuake, <http://gpquake.sourceforge.net>** An excellent port of one of the PC's most popular games, Quake. Originally written by Anders Granlund, this game's development has turned into a communal effort and is ongoing.

Other Hacks

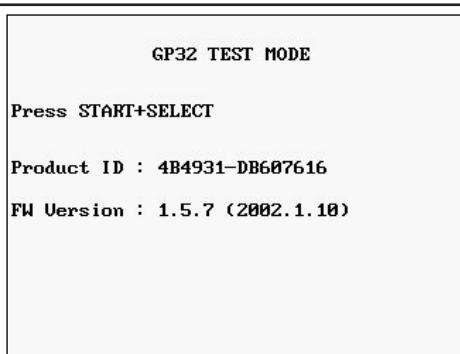
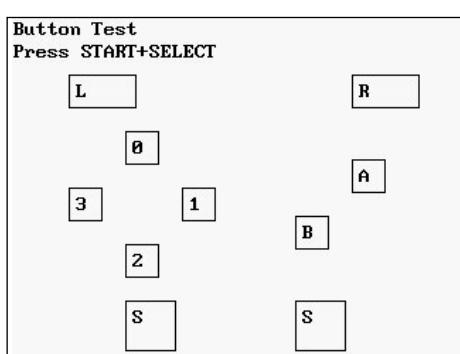
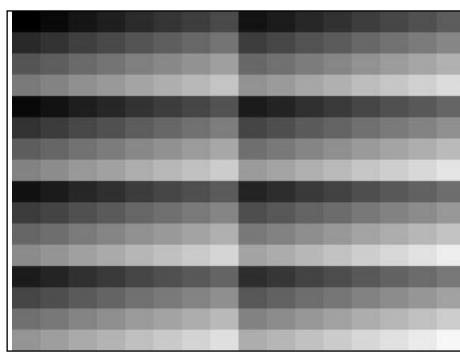
The hacks presented in this chapter are only the beginning of what you can do with your GP32. We've listed some additional hacks here; detailed instructions or pictures are available on the Internet:

- **Spray Paint Your GP32,** www.reloaded.com/nav.php?Page=articles&ID=0 A good, basic case modification for the GP32. Get rid of the boring white case color and customize it with any color you choose.
- **Start and Select Button LED Mod,** www.reloaded.com/nav.php?Page=articles&ID=2 Add LEDs to the transparent Start and Select buttons of your GP32, giving it a unique visual touch.
- **GP32 JTAG Adapter,** www.cobbleware.com/gp32/gp32jtag.html The Joint Test Action Group (JTAG) interface, also referred to as IEEE 1194.1, is an industry-standard, 4/5-wire chip-level interface used for testing and debugging purposes (system-level testing, boundary-scanning, and low-level testing of dies and components). It is commonly used during product manufacturing and testing processes. This hack provides details on building hardware and software to communicate with the GP32 processor via the JTAG interface (part of the GP32's EXT connector) and to reprogram the GP32's Flash memory.
- **GP32 FLU Dimmer Chip Hack,** www.mashmods.com/projects/gp32chip.htm Describes the creation of a custom front-light dimmer circuit for the GP32 FLU model. The circuit, connected to the GP32 with six wires, allows four levels of brightness as well as an Off setting.
- **Loading Linux onto the GP32** A popular hack with most any game console, porting Linux to the GP32 has been an ongoing challenge for many developers. The following sites are dedicated to the task and provide a wealth of information:
 - **GP32Linux,** <http://sourceforge.net/projects/gp32linux> Main page dedicated to porting Linux and associated applications to the GP32 platform.
 - **GP32Linux HOWTO,** http://voxel.dl.sourceforge.net/sourceforge/gp32linux/gp32linux_howto.html Detailed step-by-step description on how to load Linux onto your GP32. Proceed with caution—this hack is not for the faint of heart!
 - **GP32Linux gP32oPie pre-alpha,** <http://members.lycos.co.uk/fangeles/alpha/hidden/prealpha/gp32opie.html> Development notes and diary of current progress in porting Linux to the GP32.
- **The GP32 Test Mode** Though not exactly a hack, since it was intentionally designed into the system by Gamepark, the GP32 test mode provides you with a number of system-level tests of the GP32 console. Gamepark most likely uses this mode during final product

assembly, but it can be useful to hardware hackers curious about the functionality of their device (or to make sure nothing has been broken or damaged after a hack!).

To enter the GP32 test mode, turn on the device and, at the Main screen, enter the following sequence with the left and right shoulder buttons: **L, R, R, L, R, R, L, L**. You will be prompted with the Main screen of the test mode. Hold down both the **Start** and **Select** buttons to scroll through the various system tests (see Table 6.3).

Table 6.3 GP32 Test Mode Screens

Test Mode	Screenshot															
Product ID and Firmware Version	 <p>GP32 TEST MODE Press START+SELECT Product ID : 4B4931-DB607616 FW Version : 1.5.7 (2002.1.10)</p>															
Button Test	 <p>Button Test Press START+SELECT</p> <table border="1"> <tr><td>L</td><td></td><td>R</td></tr> <tr><td>0</td><td></td><td>A</td></tr> <tr><td>3</td><td>1</td><td>B</td></tr> <tr><td>2</td><td></td><td>S</td></tr> <tr><td>S</td><td></td><td>S</td></tr> </table>	L		R	0		A	3	1	B	2		S	S		S
L		R														
0		A														
3	1	B														
2		S														
S		S														
Color Test (Black, White, Color Matrix)																

Continued

Table 6.3 GP32 Test Mode Screens

Test Mode	Screenshot
Sound Test	<pre>Sound test : Press START+SELECT 261.9Hz SINE WAVE Left out : L Button Right out : R Button Stereo : A Button</pre>
SMC Test	<pre>SMC IO Test : Press START+SELECT Insert SMC into the socket</pre>
UART Test (Port 0 and Port 1)	<pre>UART Port 0 Test : Press START+SELECT Waiting to receive data..</pre>

GP32 Resources on the Web

There are a great number of resources on the Web for enthusiasts of the GP32 game system. Whether you're looking for information about your favorite games, ways to hack your system to give it capabilities never intended by the designers, or forums where you can discuss the GP32 with other fans, you're sure to find it online somewhere. Because the GP32 is commercially available and extremely

popular around the world, new Web sites are popping up all the time. The following is a list of some of our favorite GP32-related sites:

- **European GP32 Site, www.gp32eu.com** The (official) site for GP32 in Europe. Features GP32 device registration (in English!), development tools, games, utilities, and product support.
- **GP32 Xtreme, www.gp32x.com** Comprehensive site focusing on all things GP32. Features often-updated news, an active discussion forum, reviews, links, and hundreds of GP32 games, applications, and development tools.
- **GP32 World, www.gp32world.co.uk** Nice GP32 fan site containing forums, articles, guides, reviews, and a large assortment of downloads (homebrew game source code, emulators, utilities, and other applications).
- **Gamepark32.co.uk, www.gamepark32.co.uk** U.K.-based site with information on GP32 development, software, games, media players, and other information.
- **GBAX.com, www.gbax.com** Excellent site and store devoted to providing products and hacks for the GP32, Nintendo GBA, Zodiac, and other portable consoles. Also features an often-updated news section for the latest information on available tools and software. Based in the United Kingdom.
- **Gamepark Holdings, www.gamepark.com (Korean)** Main Web site for Gamepark, makers of the GP32 console.
- **Lik Sang, www.lik-sang.com** Popular Hong Kong-based online retailer of videogame consoles and game-related collectibles. Has a large inventory of products that are difficult to obtain in the United States.
- **JoyGP, <http://joygp.entware.com>** Korean-based site to purchase commercial GP32 games and obtain some freely available ones.
- **#GP32 IRC (Internet Relay Chat) Channel** Located on the EFnet server (irc.efnet.org), the #GP32 channel is dedicated to discussing all things related to the GP32 console.

Retro and Classic Systems



Nintendo NES

Topics in this Chapter:

- Introduction
- Opening the NES Console
- Replacing the 72-Pin Cartridge Connector
- Blue Power LED Modification
- Disabling the NES “Lockout Chip”
- Opening an NES Game Cartridge
- Replacing the Battery in Certain Game Cartridges
- Creating an EPROM Cartridge for Homebrew Game Development
- Homebrew Game Development
- Other Hacks
- NES Resources on the Web

Introduction

In 1985, a little more than a year after the videogame crash that decimated the industry, Nintendo announced the Nintendo Entertainment System (NES) in the United States (see Figure 7.1). The NES helped breathe new life into the videogame market, and Nintendo sold over 62 million consoles before finally discontinuing the system in 1995.

Before the NES was released in the United States, it had already existed for two years in Japan under the name of Famicom (short for *Family Computer*; see Figure 7.2). Besides being a videogame console, the Famicom provided accessories and features for computer-oriented tasks. A keyboard and modem allowed users to check e-mail and horseracing results, and a proprietary 2.8-inch floppy disk drive system provided another method to load games. (Blank floppy disks could be purchased from Nintendo, and, for a small fee, select games could be written onto the disk by vending machines located in many retail stores.) The Famicom even had its own programming language, known as *Family BASIC*, which paved the way for programmers and hobbyists to write their own homebrew games for the system. Other interesting accessories included a karaoke machine and 3-D glasses. Unfortunately, none of these features were made available in the United States, since the NES's focus was as a videogame entertainment console and not a general computing system. After lasting longer than any other videogame system in the world, production was finally stopped on the Famicom in late 2003 due to difficulty in obtaining certain electronic components.

The NES console originally retailed for \$199 and came with the now heavily recognized and franchised Super Mario Brothers game cartridge. Other package variations came later as the system gained popularity and was released in other countries. A number of accessories for the NES were available, including Robotic Operating Buddy, or R.O.B., an interactive, toy-sized robot that was controlled by light signals sent by the videogame through the television; the Zapper, a futuristic-looking gun; the Power Pad, an early foot-controlled pad (commonly seen nowadays in arcade games such as Dance Dance Revolution); and the short-lived Power Glove, a weak attempt at a virtual reality controller.

Nintendo ultimately released two versions of the NES in the United States. The first was affectionately called the Toaster (see Figure 7.1) because of its boring, boxy appearance that somewhat resembles a traditional toaster you'd find in any kitchen. In 1993, 10 years after the original release of the system in Japan, a more stylized, less expensive version of the Famicom was released there. A U.S. release, referred to as the NES 2 or Top Loader, followed shortly thereafter (see Figure 7.3). The Top Loader system had some fine upgrades from the Toaster, including a cartridge connector interface that was much more reliable. (The original NES release had a loading mechanism that constantly created connection problems between the game cartridge and the console; the “Replacing the 72-Pin Cartridge Connector” hack in this chapter presents a fix.) The Top Loader removed the composite video output from the back of the unit, much to the chagrin of many gamers, and included two cosmetically modified controllers called Dog Bones because of their obvious likeness to dog bones. For a more complete history of NES, take a look at *Game Over: Press Start to Continue*, written by David Sheff (ISBN 0-9669617-0-6).

Figure 7.1 The Original Nintendo Entertainment System, Also Known as the Toaster



Figure 7.2 The Original Famicom Console, Released in Japan



Figure 7.3 The Updated Release of the Nintendo Entertainment System, Also Known as the NES 2 or Top Loader



For this chapter, we focus solely on using the Toaster for our hacks. Our first four hacks involve modifying the NES console. The last three hacks involve modifying NES game cartridges.

Opening the NES Console

The first few hacks in this chapter require that you open the Toaster. Opening the system is simple, but you'll need to keep track of more than 20 screws that you'll remove from the console. Before starting the hack, set aside a clean area where you can place all the screws so they don't get lost.



Preparing for the Hack

The only tool required for this hack is a standard size Phillips screwdriver.

Performing the Hack

Perform the following:

1. To open the NES console, which will give you access to all the internal circuitry, first flip the case upside down and remove the six screws holding the case together, as denoted in Figure 7.4.

Figure 7.4 The Underside of the Nintendo NES, Showing Screw Locations



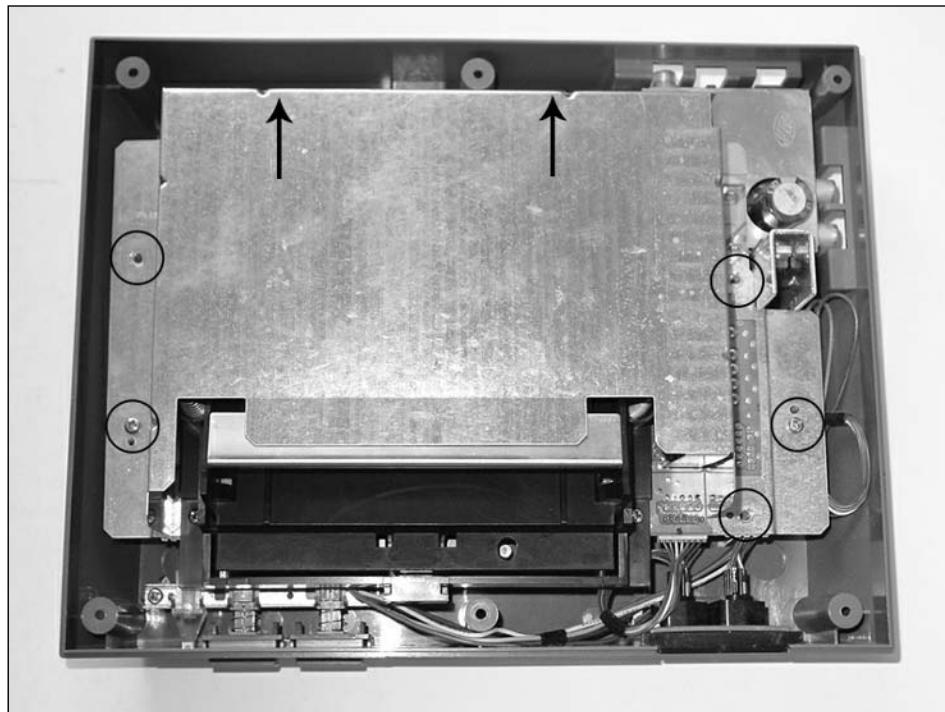
2. The case consists of two plastic pieces—the top and the bottom. After you remove the screws, carefully lift the bottom of the NES straight upward, separating it from the top (see Figure 7.5).

Figure 7.5 Separate the Bottom and Top Halves of the Console

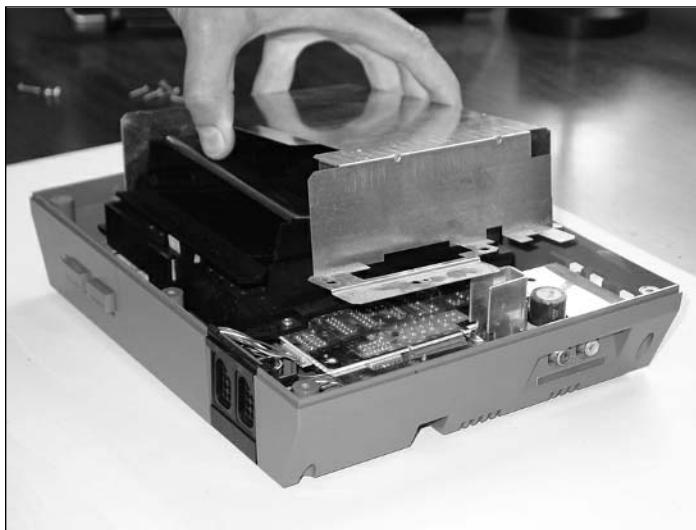


3. Flip the bottom of the system back to its upright position and set the top piece of the case aside. You won't need it until you've finished your hacks and it's time to put the system back together.
4. Next, remove the seven screws that fasten the RF shield on top of the main circuit board, as denoted in Figure 7.6. There are two screws on the back side, two screws on the left side, and three screws on the right side.

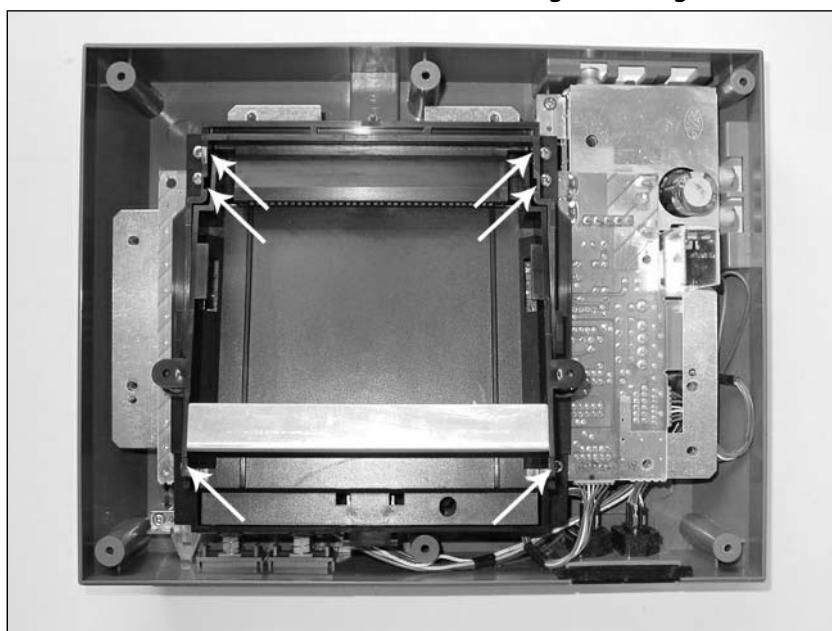
Figure 7.6 Remove the Seven Screws from the RF Shield



5. After you remove the screws, simply lift the RF shield off the circuit board (see Figure 7.7). You might have to wiggle the shield slightly as you're lifting it to ensure that it isn't catching on any components.

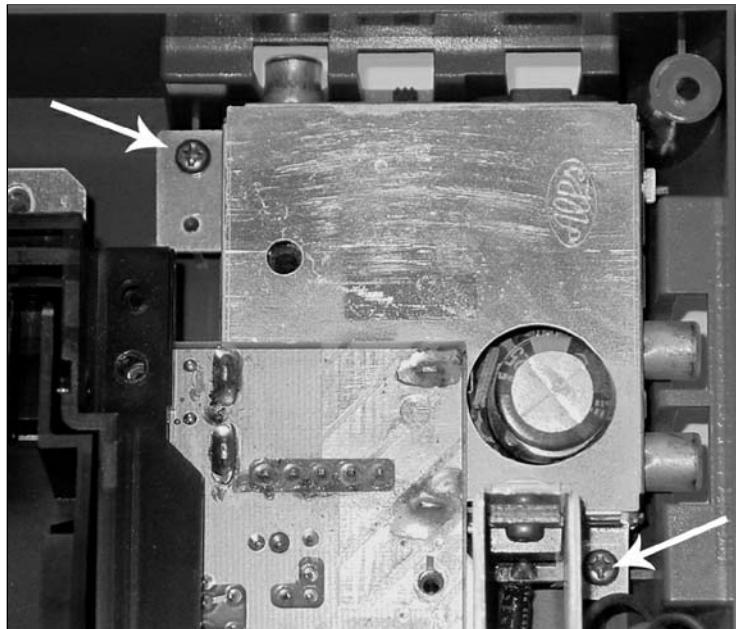
Figure 7.7 Remove the RF Shield from the Console

6. Remove the six screws that fasten the cartridge loading mechanism to the main circuit board, as denoted in Figure 7.8. There are three screws on each side of the mechanism. The middle screw (second from the top) on each side is longer, so make a note of that for when you reassemble everything after your hack.

Figure 7.8 Remove the Six Screws from the Cartridge-Loading Mechanism

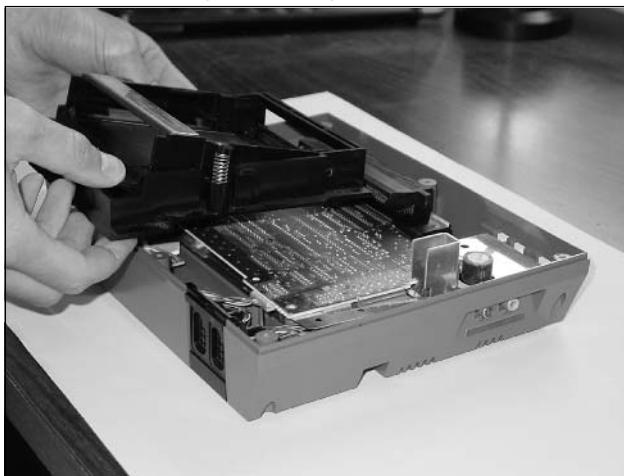
7. Now remove the two screws that secure the RF demodulator to the bottom NES housing (see the close-up in Figure 7.9).

Figure 7.9 Remove the Two Screws from the RF Demodulator



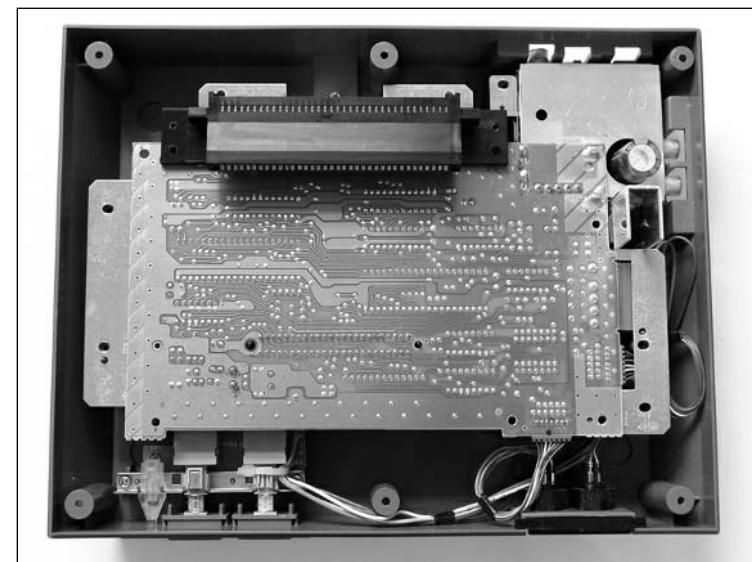
8. With the screws removed, pull the cartridge-loading mechanism toward the front of the system and lift it up and away from the console (see Figure 7.10). You might have to wiggle the cartridge tray slightly to remove it.

Figure 7.10 Remove the Cartridge-Loading Mechanism



9. Your system should now resemble the system shown in Figure 7.11, which will serve as the starting point for the other hacks and modifications in this chapter.

Figure 7.11 Opened NES System Ready for Hacking



To put the system back together, simply follow the disassembly steps in reverse order:

1. First, reattach the cartridge-loading mechanism and ensure that everything is sitting flush inside the bottom NES housing.
2. Next, screw the two screws back into the RF demodulator.
3. Screw the six screws back into the cartridge-loading mechanism, making sure that the two long ones are the center screw of each side of three.
4. Place the top RF shield over the circuit board, and secure it into place with the seven screws.
5. Finally, attach the top half of the NES plastic housing and screw in the six screws to hold the enclosure together.

Your NES should now be reassembled!

Replacing the 72-Pin Cartridge Connector

Remember trying to blow the dust out of your cartridges in hopes that they would work in your NES? Or even blowing into your system or strategically placing the cartridge into the cartridge tray to avoid the dreaded blue flashing screen? Years later, you can know the truth: The most common

failure of the first U.S. version of the NES, the Toaster, was due to the 72-pin cartridge connector inside the console becoming worn and stretched out over time.

The Toaster was notorious for failures, but they were a fact of life that most gamers just dealt with. The second U.S. release of the NES, the Top Loader, used a different cartridge-loading system and a different style of 72-pin connector. The Top Loader rarely experienced the same type of poor connections on the cartridge connector as the Toaster. For those lucky gamers who obtained the more rare Top Loader system, they didn't have to deal with the misery of the bad connector.

This hack will guide you through replacing your old, worn-out cartridge connector in your NES Toaster with a brand-new one, breathing new life into your system. Finally, you can enjoy playing your NES games without worrying about the cartridge coming loose or the game simply not working.



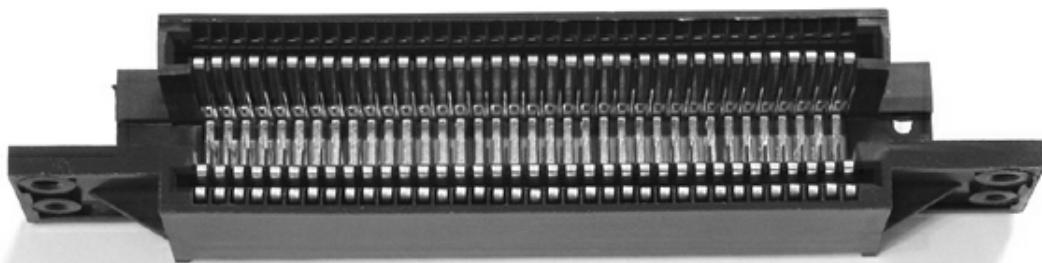
Preparing for the Hack

This hack only requires:

- A standard size Phillips screwdriver
- A new 72-pin cartridge connector (see Figure 7.12)

The 72-pin cartridge connector can be purchased from many videogame shops and from various suppliers on eBay for \$5 to \$10. Alternatively, searching for *NES Connector* on eBay will provide you with many options.

Figure 7.12 A New 72-Pin Cartridge Connector for the NES Toaster

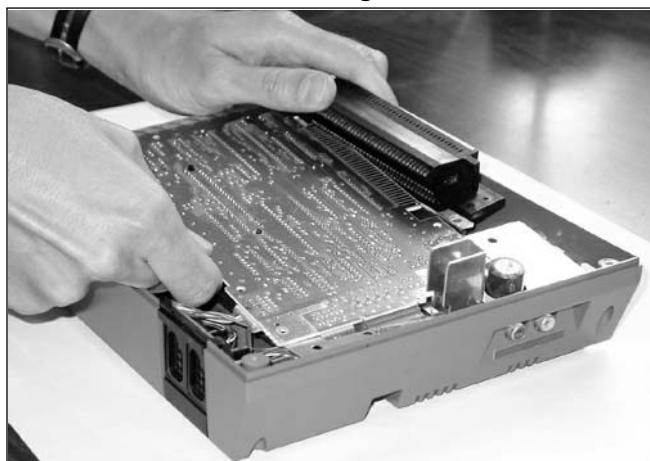


Performing the Hack

Perform the following:

1. First, you need to open your NES system as described in the “Opening the NES Console” section earlier in this chapter.
2. Once the system is opened, the next step is to remove the old 72-pin connector. This can be done by holding the connector in one hand and gently wiggling it back and forth while pulling it away from the circuit board (see Figure 7.13). You might have to lift the circuit board slightly for the connector to have enough clearance to be removed.

Figure 7.13 Remove the Old 72-Pin Cartridge Connector



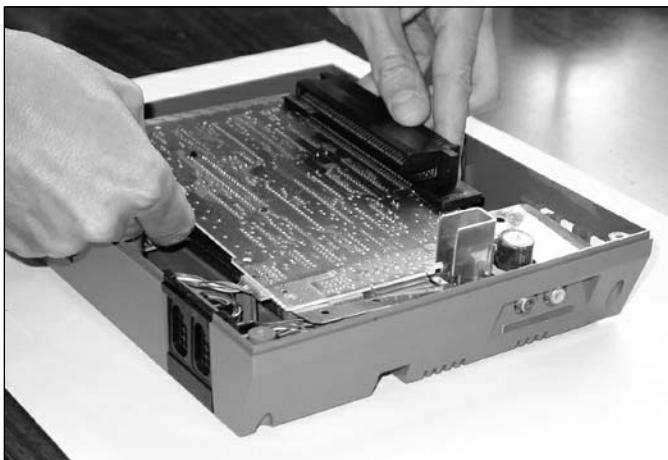
With the old 72-pin cartridge connector removed, your system should resemble the one in Figure 7.14.

Figure 7.14 The Old 72-pin Cartridge Connector Is Successfully Removed



- Now all we have to do is attach the new 72-pin cartridge connector onto the NES circuit board. This can be done by aligning the cartridge connector with the board and sliding it into place (see Figure 7.15). The cartridge connector should have a tight fit with the circuit board; you might need a little bit of force to make sure it is fully seated.

Figure 7.15 Attach the New 72-Pin Cartridge Connector



- Congratulations! Your new 72-pin cartridge connector should be installed and ready to go. Now reassemble the system as described in the previous “Opening the NES Console” section and enjoy your updated system. When you insert a game cartridge into the system, there should be a nice, snug fit. Look, Mom, no flashing blue screen!

If your system still blinks after replacing the cartridge connector, the problem is likely due to dirty contacts on the game cartridges. Using your favorite cartridge-cleaning product, thoroughly clean the contacts of your cartridges and try again. A popular solution is to use a Q-Tip with Windex or a mixture of 50% isopropyl alcohol and 50% water, rubbing the metal cartridge contacts thoroughly until the dirt has been removed.

Blue Power LED Modification

The Nintendo NES Toaster features a red LED that is illuminated when the system power is on. This hack explains how you can replace this stock LED with a blue LED to add a unique touch to your NES console. This hack only works on the first U.S. version of the NES. The second U.S. version, the Top Loader, does not have a power LED indicator.



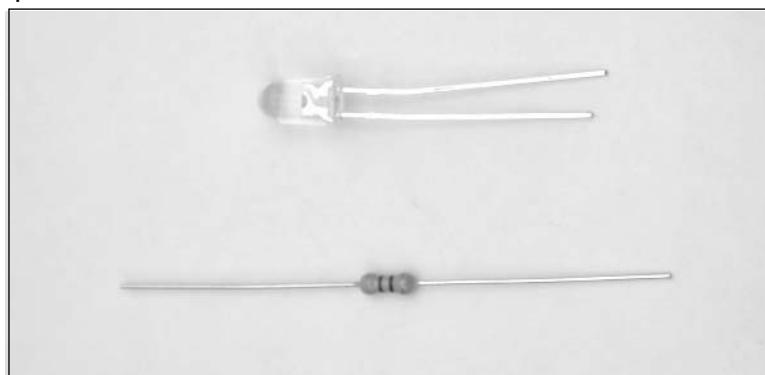
Preparing for the Hack

For this hack, you'll need two components (see Figure 7.16):

- A blue LED (2600mcd, 3.7V, 20mA, Radio Shack part #276-316)
- An 820 ohm, 5% resistor

We'll use a blue LED with a forward voltage of 3.7V and a brightness of 2600mcd @ 20mA coupled with an 820 ohm current limiting resistor. You can experiment with different values of resistance if you'd like a brighter or dimmer LED. The original resistor installed in the NES is 220 ohms, which, if paired with the new blue LED, results in significantly higher light output and is pretty overwhelming. So, increasing the resistance will reduce the brightness of the power LED. If you can't find an 820 ohm resistor, you could use a 470 ohm (Radio Shack part #271-1317) and a 330 ohm (Radio Shack part #271-1315) connected in series, which will give you an equivalent resistance of 800 ohm. (See Appendix A, "Electrical Engineering Basics," for details on how resistors work.)

Figure 7.16 Required Parts



The tools required for this hack are:

- A Phillips screwdriver, standard size
- A soldering iron
- Solder sucker or solder braid
- Wire cutters

Performing the Hack

Perform the following:

1. First, you need to open your NES system as described in the “Opening the NES Console” section earlier in this chapter. This hack requires a modification to both the top and bottom pieces of the case. We’ll work on the top piece first.
2. With the system opened, take the main NES circuit board and flip it over the case to the right of the console, as shown in Figures 7.17 and 7.18.

Figure 7.17 Flip Over the NES Circuit Board

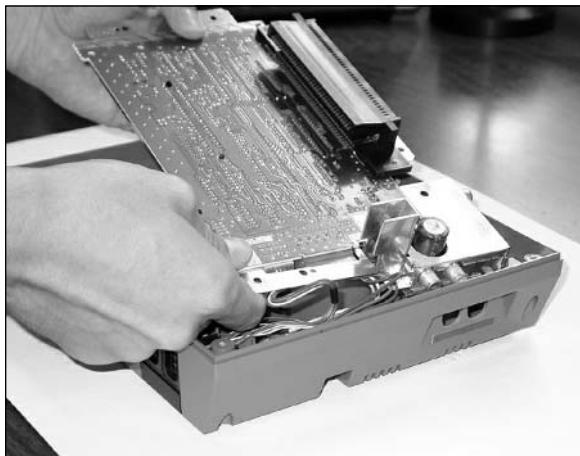
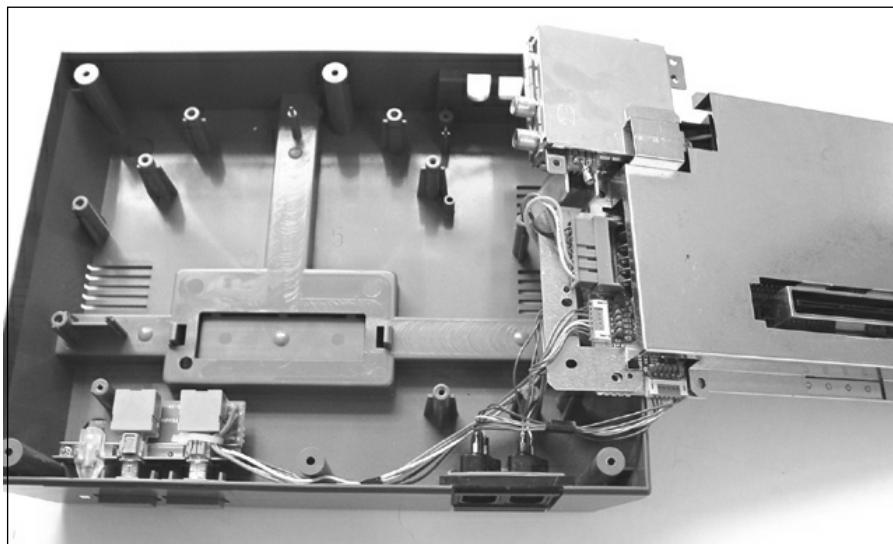
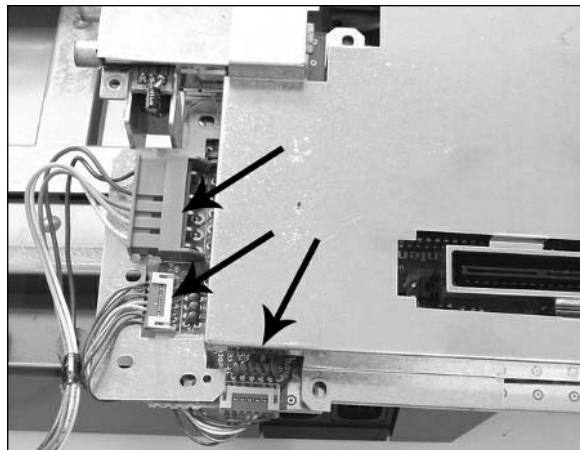


Figure 7.18 NES Console with the Circuit Board Flipped Over



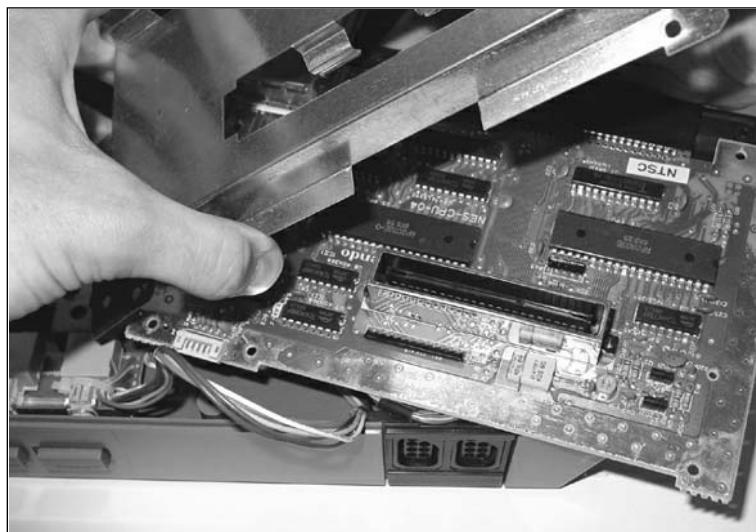
3. Next, it is necessary to remove the three connectors from the main circuit board, as denoted in Figure 7.19. When removing the three connectors, be sure to pull them by their plastic headers, *not* from the wires (or else you could pull the wires out of the connector). Removing the connectors will allow you better access to the circuit board without having to balance it precariously on the edge of the NES bottom housing.

Figure 7.19 Remove the Three Connectors



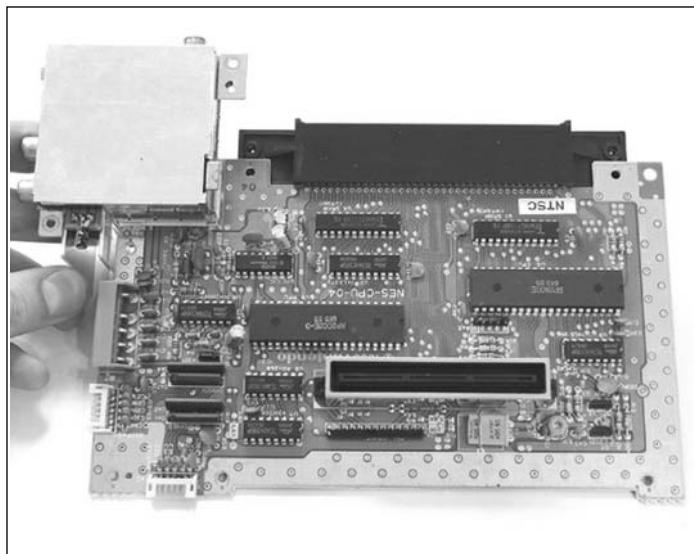
4. With the connectors unplugged, you can now remove the RF shield from the top of the NES circuit board (see Figure 7.20). Place the shield aside; you won't need it again until it's time to put the system back together.

Figure 7.20 Remove the RF Shield



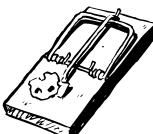
5. The NES circuit board should now be free from the rest of the console and should resemble the one shown in Figure 7.21. At this point, the disassembly is complete, and we can continue with the hack.

Figure 7.21 NES Circuit Board Removed from the Console



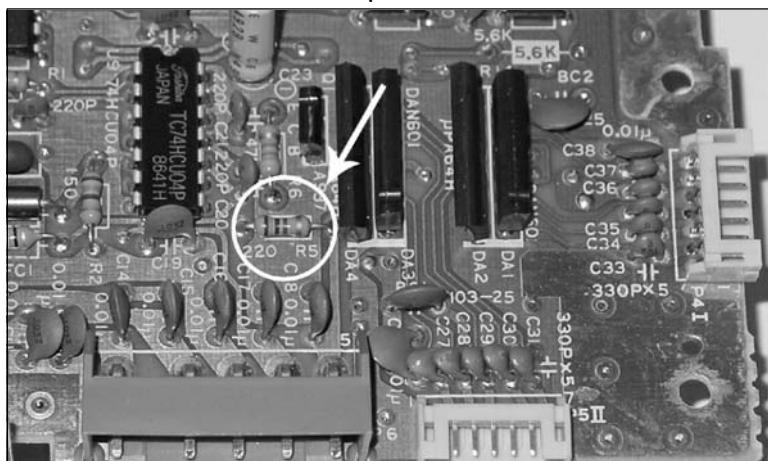
6. Now we need to remove the old resistor from the NES circuit board and replace it with the new one. Figure 7.22 shows a close-up of the board with the resistor to remove. Unsolder the two connections of the old resistor, which is denoted by “220 R5” on the silkscreen. Carefully pull the resistor off the board and discard it.

WARNING: HARDWARE HARM



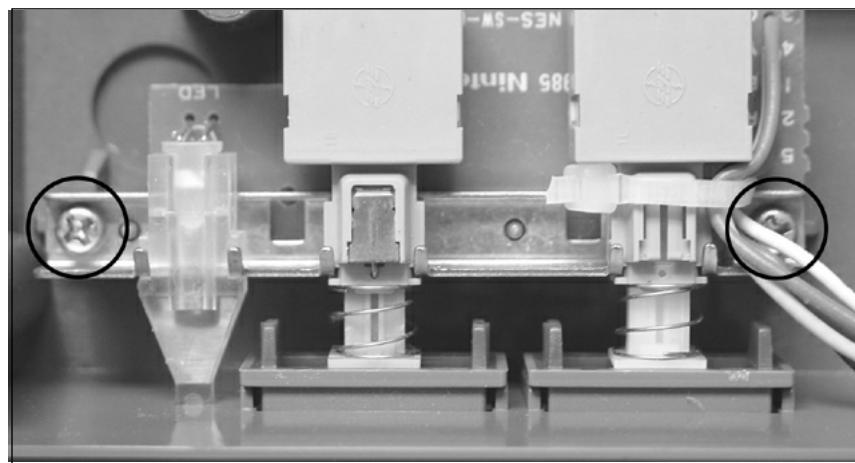
Be careful when removing the resistor from the board—you don't want to pull up the solder pads from the circuit board. Try to remove as much solder as possible from the joints on the bottom of the board, and then carefully pull the resistor up from the board while heating the connections.

7. Next, put the new resistor into the location of the old one and solder it into place. The new resistor can be placed in either orientation—the polarity does not matter.

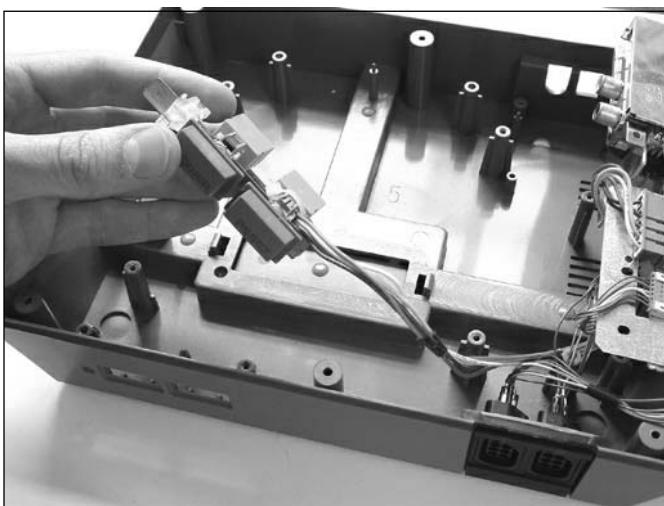
Figure 7.22 Remove the Resistor and Replace with the New One

The next few steps involve removing and replacing the LED. Redirect your attention to the bottom housing of the NES console.

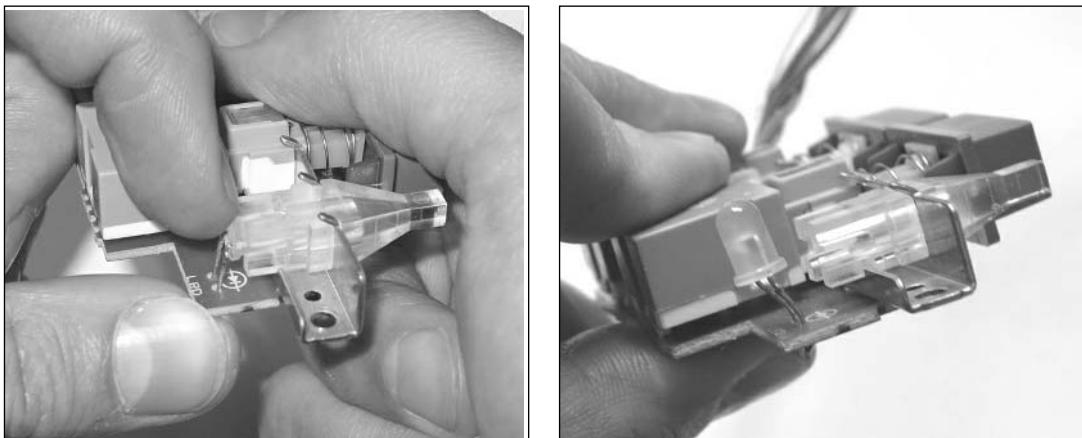
8. On the front-left corner of the system, you will see a small circuit board containing an LED and two buttons (for the Power and Reset buttons of the NES console). Unscrew the two screws that fasten this small circuit board to the bottom housing (see Figure 7.23).

Figure 7.23 The NES LED and Button Circuit Board

9. With the two screws removed, lift the small circuit board away from the housing (see Figure 7.24). This will give you access to the top and bottom of the circuit board.

Figure 7.24 Remove the Circuit Board from the NES Housing

10. Next, we need to remove the original LED from the light pipe. To do so, carefully use your fingernail to pull it out of the pipe (see Figure 7.25).

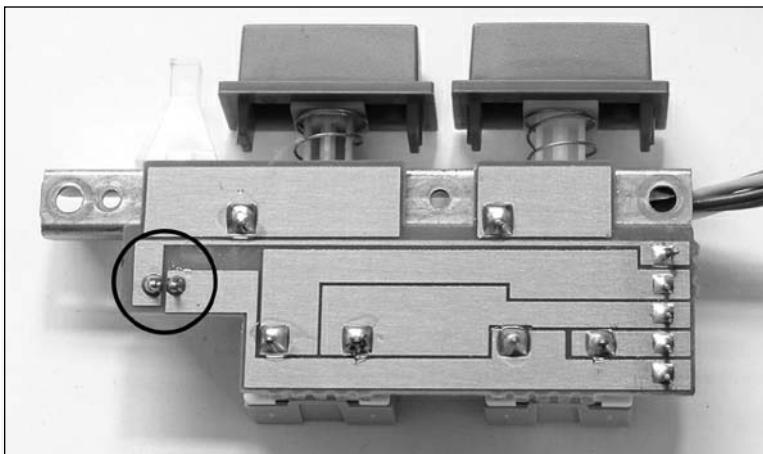
Figure 7.25 Remove the LED from the Light Pipe, During and After

11. Now we need to remove the original LED from the small circuit board. Figure 7.26 shows the solder side of the circuit board with the two pads of the LED. Unsolder the two connections, carefully pull the LED off the board, and discard it.

WARNING: HARDWARE HARM

Be careful when removing the LED from the board—you don't want to pull up the solder pads from the circuit board. Try to remove as much solder as possible from the joints on the bottom of the board, and then carefully pull the LED up from the board while heating the connections.

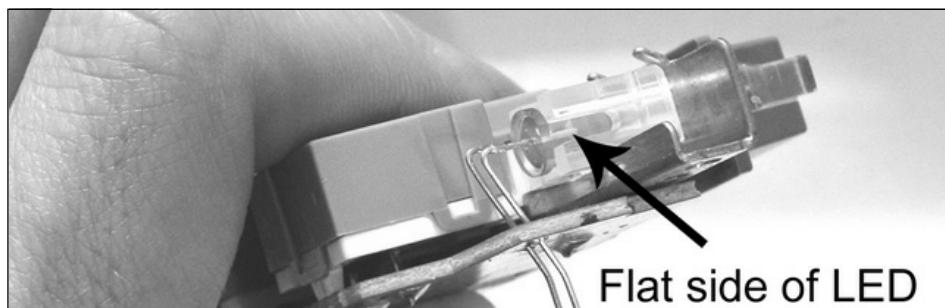
Figure 7.26 Unsolder and Remove the LED



Now it's time to put in the new blue LED. When replacing the new LED in the circuit board, make sure that the LED is oriented properly. The flat side of the LED should align with the cathode marking of the diode silkscreened on the board (the vertical line at the tip of the triangle), which is closest to the right edge of the board when you're looking down at the components.

12. First, bend the leads of the LED 90 degrees downward.
13. Next, insert the leads of the LED into the proper holes on the circuit board.

Figure 7.27 New LED Installed onto Circuit Board



14. Then bend the LED backward slightly while fitting it into place in the light pipe. Take care not to put too much stress on the leads of the LED to prevent damaging it. Your circuit board should resemble the one in Figure 7.27.
15. Finally, solder the new blue LED onto the board and clip off the excess leads of the LED.
16. You can now reassemble your NES by screwing the small LED/button circuit board back into the front panel and replacing the top RF shield. When reconnecting the three connectors to the main circuit board, take care to ensure that the Controller I and Controller II connectors are plugged into the correct headers. (If the board is oriented as in Figure 7.21, Controller I is the connector on the bottom of the board, denoted by “P4 I” on the silkscreen, and Controller II is the connector on the left side of the board, underneath the button connector, denoted by “P5 II” on the silkscreen.) Flip the circuit board back over and place it onto the mounting posts of the NES bottom housing, making sure that no wires are being pinched underneath.
17. To complete the reassembly, follow the steps as described in the previous “Opening the NES Console” section, and enjoy the new look of your system (see Figure 7.28)!

Figure 7.28 Blue LED Happiness on the Nintendo NES Toaster



Under the Hood: How the Hack Works

This hack consisted of two steps: replacing a resistor and replacing the LED. If you replaced only the LED and not the resistor, the blue LED would appear much brighter than the original red LED. This is because the original current-limiting resistor (R5 on the main circuit board) is 220 ohm, allowing approximately 15mA to flow through the LED.

Because blue LEDs are inherently brighter than red, given the same amount of current, we need to increase the value of the current-limiting resistor to reduce the amount of current going to the LED. Typically, a red LED has a forward voltage of 2V, a green LED has a forward voltage of 2.2V, and

a blue LED has a forward voltage of 2.7V to 3V—different-colored LEDs have different voltage drops, so it is not often that you can just replace one color LED with another and assume you will have the same brightness. Having the power indicator LED too bright can draw your attention away from the TV screen when you are playing games.

So how did we choose the new value of the current-limiting resistor? Well, with an application such as this (which is highly subjective), you often need to experiment to find a brightness that is appropriate. After some trial and error with various resistance values, the 820 ohm value provided a nice, bearable illumination for the LED.

The NES console supplies 5 volts to the LED. The resistor is placed in series with the LED to act as a current limiter and to let only the desired current pass to the LED. This is done to protect the LED from exceeding its maximum current specification (in this case, 20mA). By adjusting the value of the resistor, we adjust the amount of current going to the LED, which in turn changes the brightness of LED illumination.

Armed with the specifications of the blue LED (in our case, a forward current of 20mA and a forward voltage of 3.7V) and our new resistor value, we can calculate the amount of current flowing through the LED using Ohm's Law, which states:

$$\text{Voltage} = \text{Current} * \text{Resistance}, \text{ or } V = I * R$$

To calculate current, we use the formula as follows:

$$\text{Current} = \text{Voltage} / \text{Resistance}, \text{ or } I = V / R$$

Plugging in the values for the known voltage and resistance, we get:

$$I = (5V - 3.7V) / 820 \text{ ohms} = 0.00158A = \text{approximately } 1.6mA$$

Using this formula, we can see that we are driving the LED at just about 1.6mA, which is only a fraction of the maximum allowable current of the LED, significantly reducing its brightness. Feel free to experiment with different values of resistance to increase or decrease the light output of the LED.

Disabling the NES “Lockout Chip”

The NES “Lockout Chip” is a security mechanism that Nintendo used to maintain its exclusivity on cartridge manufacturing and to control game distribution (a form of territorial protection) for the NES. Also referred to as the “10NES,” the mechanism is essentially a “lock-and-key” design in which the Lockout Chip inside the NES communicates with an identical Lockout Chip inside the game cartridge to authorize the game to be played. This was meant to prevent unlicensed and foreign games from being played on the console (for example, NES systems released in the United States were only allowed to play games released in the United States, European systems were only allowed to play games released in Europe, and so on). At least four different types of Lockout Chip were used in U.S., U.K., Europe, Italy, and Hong Kong consoles. The Lockout Chip can be considered an early form of region encoding made popular by DVD discs and players.

Nintendo had very strict guidelines as to the quality and types of games that could be produced and sold for the NES. The familiar gold “NES Seal of Approval” was printed on all Nintendo-licensed game cartridges, manuals, and boxes. Adult content or gratuitous violence was unacceptable.

Additionally, Nintendo required third-party developers to pay the company for the right to publish games for the system. Nintendo, which took a portion of the profits, then manufactured the games. The Lockout Chip prevented third-party developers from manufacturing the game cartridges themselves; instead they were forced to go through Nintendo and adhere to Nintendo's guidelines.

It wasn't surprising that the functionality of the 10NES security mechanism was eventually reverse engineered by a number of companies (Tengen, which was an offshoot of Atari, Color Dreams, and Wisdom Tree, to name a few), which began producing unlicensed NES games, many of which remain rare to this day since distributors that sold unlicensed games, obviously, faced strong opposition from Nintendo.

Details of the Lockout Chip functionality can be found in U.S. Patents # 4,799,635, "System for Determining Authenticity of an External Memory Used in an Information Processing Apparatus," and # 5,070,479, "External Memory Having an Authenticating Processor and Method of Operating Same." Macronix, Inc., actually filed a patent detailing a method for emulating or bypassing the 10NES mechanism with a lower-cost (and unauthorized by Nintendo) solution (U.S. Patent # 5,004,232, "Computer Game Cartridge Security Circuit"). These, and all other patents, can be viewed and downloaded for free at www.us-patent-search.com.

The following hack will disable the functionality of the NES Lockout Chip, thus allowing foreign games and unlicensed third-party U.S. games to be played on your console. However, a few unlicensed game cartridges (such as Firehawk and Quattro Adventure, written by Codemasters and published by Camerica) require that the Lockout Chip be enabled, so they won't work after this hack is done. Additionally, this hack only works on the first U.S. version of the NES, the Toaster. The second U.S. version, the Top Loader, does not have the Lockout Chip installed. It's unclear why Nintendo removed the security mechanism from the Top Loader.

Let's get started!



Preparing for the Hack

Preparation for this hack is easy. The only part required for this hack is a 2-inch length of 22-26AWG wire, which will be used to connect two pins together on the NES circuit board.

The tools required for the hack are:

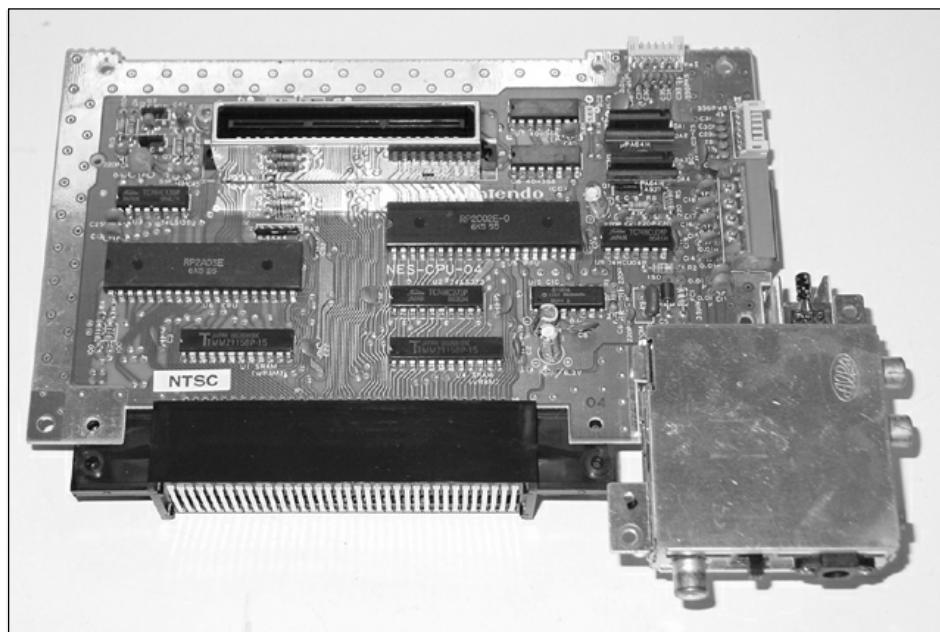
- A Phillips screwdriver, standard size
- A soldering iron
- Wire cutters

Performing the Hack

Perform the following:

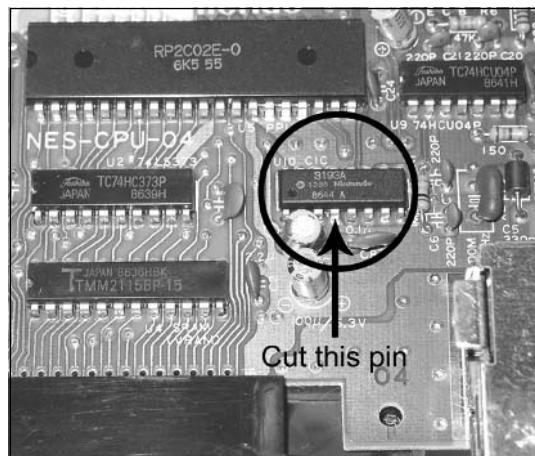
1. First, you will need to gain access to the NES circuit board. To do this, open your NES system as described in the "Opening the NES Console" section earlier in this chapter and then follow Steps 1 through 5 in the "Blue Power LED Modification" hack. The NES circuit board should now be free from the rest of the console and should resemble the one shown in Figure 7.29.

Figure 7.29 NES Circuit Board Removed from the Console

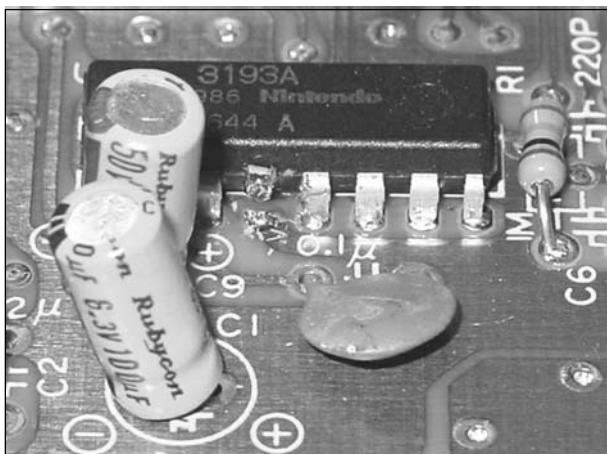


2. Now, using the wire cutters, clip pin 4 of U10 (denoted by “U10 CIC” on the silkscreen; see Figure 7.30) as close to the circuit board as possible.

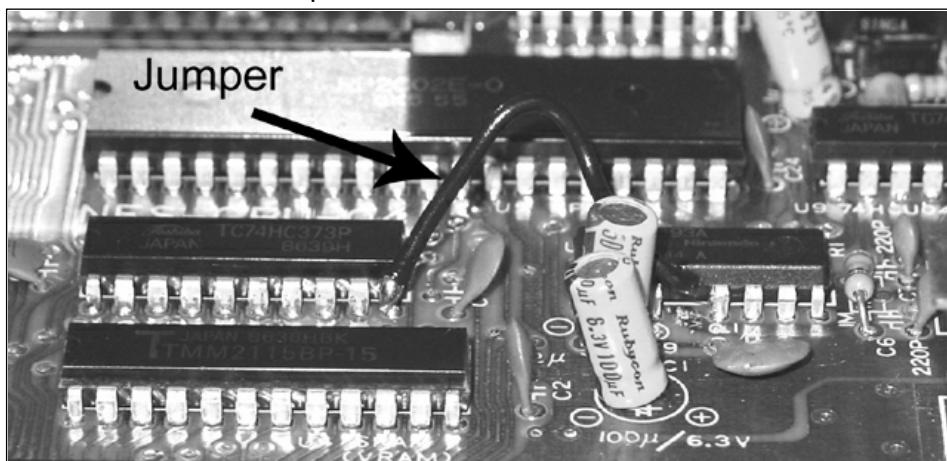
Figure 7.30 Close-Up of the NES Circuit Board Showing the Lockout Chip



3. Bend the cut lead (which should still be attached to the chip) up and away from the circuit board, as shown in Figure 7.31.

Figure 7.31 Pin Successfully Cut and Lifted from the Lockout Chip

- Next, strip 1/8 inch of the plastic shielding from each end of the 2-inch length of wire. Solder one end to the pin 4 lead that was just cut, and solder the other end to pin 10 of U2 (denoted by “U2 74LS373” on the silkscreen). With the jumper wire soldered into place, your system should resemble the one shown in Figure 7.32.

Figure 7.32 Solder the Jumper Wire Between the Two Pins

- You can now begin to reassemble your NES by replacing the top RF shield and reconnecting the three connectors to the main circuit board. When doing so, take care to ensure that the Controller I and Controller II connectors are plugged into the correct headers. (If the board is oriented as in Figure 7.21, Controller I is the connector on the bottom of the board, denoted by “P4 I” on the silkscreen, and Controller II is the connector on the left

side of the board, underneath the button connector, denoted by “P5 II” on the silkscreen.) Flip the circuit board back over and place it onto the mounting posts of the NES bottom housing, making sure that no wires are being pinched underneath.

6. To complete the reassembly, follow the steps as described in the previous “Opening the NES Console” section.

The hack is complete, and you should now be able to enjoy using your NES without the annoyances of the Lockout Chip! To check whether the Lockout Chip has really been disabled, power on the console without a cartridge inserted into it. There should just be a blank screen and not the familiar flashing blue screen.

Enjoy your “universal” NES!

Optional: Adding a Switch

Because there are certain games that will not work unless the Lockout Chip is *enabled*, it might be useful to have a way to toggle between enabling and disabling the chip. This optional step, requiring a few modifications to the previous hack, will allow you to do so with a flip of a switch. You will need:

- An SPDT toggle switch (such as the SPDT Sub-Mini Toggle Switch, Radio Shack #275-613, or the SPDT Micromini Toggle Switch, Radio Shack #275-625)
- Three short lengths of 22-26AWG wire

Before starting with this optional section, remove the jumper wire (shown in Figure 7.32) if you added one:

1. Connect the middle lead of the switch to pin 4 of U10 (the Lockout Chip).
2. Next, connect one of the other leads of the switch to the pad of pin 4 on the NES circuit board. This step will be a little tricky because there will not be much area to solder the wire onto, as you can see in Figure 7.31.
3. Finally, connect the remaining lead of the switch to pin 10 of U2 (denoted by “U2 74LS373” on the silkscreen).
4. With the switch installed, one direction will enable the Lockout Chip by connecting the cut lead back onto the circuit board (as in the original NES configuration) and the other direction will disable the Lockout Chip by pulling the pin to ground (see the next section for details).

You can now enable or disable the Lockout Chip as you wish!

Under the Hood: How the Hack Works

The NES Lockout Chip (denoted by “U10 CIC” on the NES circuit board) is part of a security mechanism that Nintendo used to maintain its exclusivity on cartridge manufacturing for the NES. The Lockout Chip inside the NES communicates with an identical Lockout Chip inside the game cartridge before allowing the game to play. This system was meant to prevent unauthorized games produced and manufactured by third parties from being played on the NES.

Depending on the state of pin 4 of the Lockout Chip, the chip is configured to act as either the “lock” (when set to +5V, as done inside an unmodified NES console) or the “key” (when set to 0V or ground, as done inside an NES game cartridge). Since pin 4 of the Lockout Chip in the console is normally pulled high, the lock mechanism is enabled. Clipping this pin from the circuit board and connecting it to ground will configure the Lockout Chip to act as the key. As described in Nintendo’s patent, when both the console and game cartridge are set to the key state, an unstable state takes place and no operations are performed at all. This means that the security mechanism is effectively disabled.

Although this hack will work by simply clipping pin 4 and leaving it floating (in other words, not connecting it to ground), doing so is not recommended, because there could be intermittent problems due to the fact that the signal is not set to a steady state. For only a few minutes of extra time to add the jumper wire, the hack can be done correctly to ensure the system’s proper operation. Depending on your preference, instead of soldering the one side of the wire to pin 10 of U2, you can also solder the wire to pins 11, 12, 13, 14, or 15 of U10 (the Lockout Chip), because they are all ground connections.

Opening an NES Game Cartridge

The last few hacks in this chapter require modifying NES game cartridges. To do so, you need to open the cartridge and gain access to the internal circuitry. This hack will guide you through the simple process of opening an NES game cartridge.



Preparing for the Hack

To complete this hack, you will need:

- An NES cartridge that you want to open
- A jeweler’s size flathead screwdriver or an NES 3.8mm security bit (depending on the cartridge used)

NES game cartridge housings are screwed together with either three or five screws. If the cartridge you want to open uses five screws (see Figure 7.33), the only tool you will need is a jeweler’s size flathead screwdriver. If the cartridge you want to open uses three screws (see Figure 7.34), the only tool you will need is an NES 3.8mm security bit (also known as a *gamebit*). The 3.8mm security bit (close-up shown in Figure 7.35) fits onto specially shaped screws used to keep the cartridge together and is also used on some Sega game cartridges and systems. The bit is available from MCM Electronics (www.mcmelectronics.com, part #22-1145) as well as many online videogame retail stores and eBay (do a search for *NES Security Bit*).

Figure 7.33 Back of an NES Game Cartridge, Five-Screw Version



Figure 7.34 Back of an NES Game Cartridge, Three-Screw Version

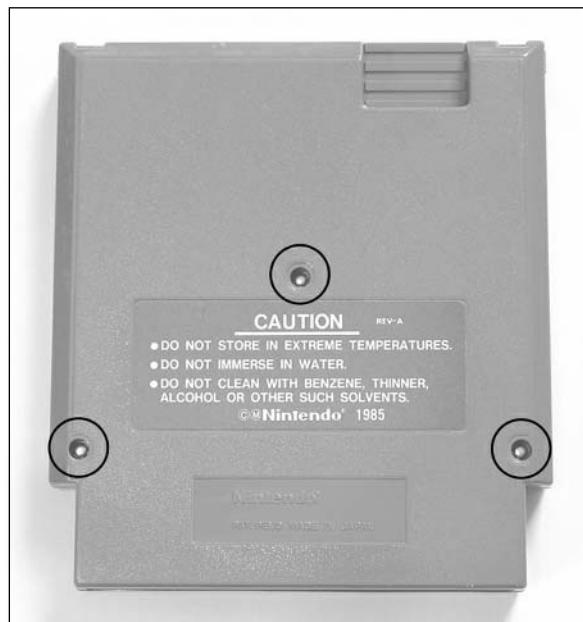
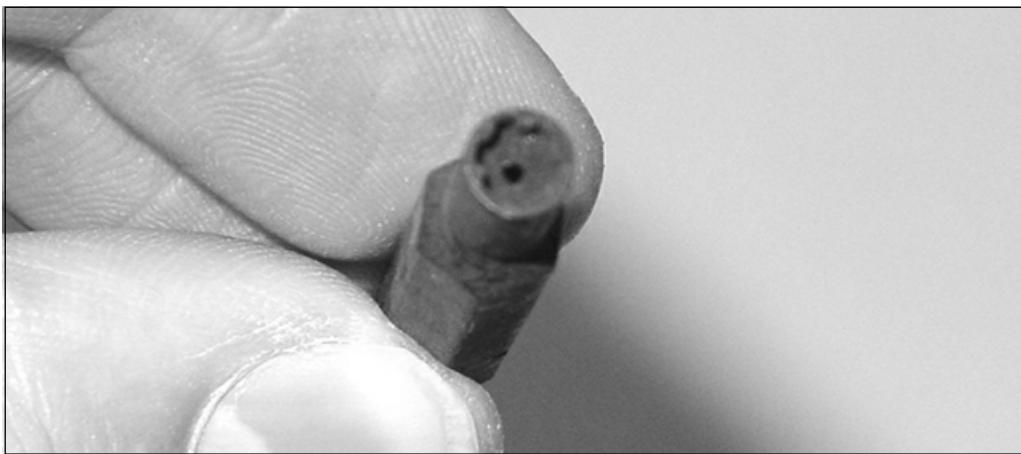


Figure 7.35 Close-Up of the NES 3.8mm Security Bit

Performing the Hack

Perform the following:

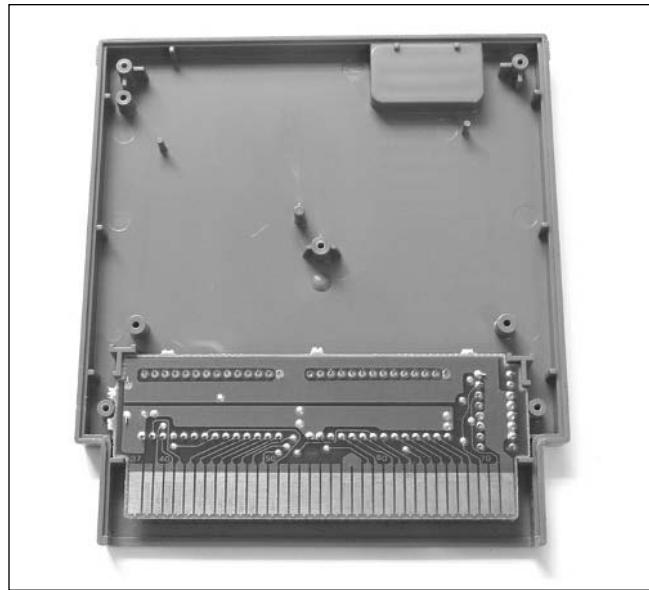
1. The first and only step in this hack is to remove the screws from the back of the game cartridge (see Figure 7.36). The screws are very small, so place them in a safe place after you remove them so you don't lose them.

Figure 7.36 Remove the Screws from the NES Game Cartridge

Your game cartridge will now be opened, which will serve as the starting point for the other hacks and modifications in this chapter. Your cartridge should resemble the one shown in Figure 7.37.

Note that there are many different types of circuit boards used in NES games. Different games use different boards, depending on their program size and functionality, so your particular game might not resemble this image exactly. For more information on the different types of game cartridge circuit board types, see the “Creating an EPROM Cartridge for Homebrew Game Development” section later in this chapter.

Figure 7.37 NES Game Cartridge Successfully Opened



To reassemble the game cartridge, simply place the circuit board back into the top plastic housing (see Figure 7.37), replace the bottom plastic housing, and screw the two pieces together. Piece of cake!

Replacing the Battery in Certain Game Cartridges

A number of Nintendo’s NES games provided a method to save your game data to allow you to continue playing the game at a later date. This was particularly useful for role-playing games (RPGs) that required dozens of hours (or days!) to complete and couldn’t realistically be done in one sitting. Games such as *The Legend of Zelda*, *Zelda II: Link’s Adventure*, *Dragon Warrior I* and *II*, *Final Fantasy*, *Ultima: Exodus*, and *Tecmo Super Bowl* included the feature.

NES game cartridges with this battery-backup feature usually (but not always) have a special gold sticker on the back of the cartridge, which states that “The Game Pak contains batteries” and lists some operating instructions (see Figure 7.38).

Figure 7.38 Back of an NES Game Cartridge, Battery-Backup Version

Saving such games and data required the use of a battery internal to the cartridge that would supply power to a small Random Access Memory (RAM) device. Such RAM is volatile, meaning that the contents would be lost if power were removed. So, by having an internal battery, the data can remain active and was designed to do so for approximately five to 10 years. The problem is that these games were made and released in the mid- to late-1980s—nearly 20 years ago. It is only natural that the batteries will lose their charge after such a long time period, just as with regular batteries that you use in everyday life.

This hack will show you how to replace the old battery in the NES cartridge with a brand-new battery, which will save your games for another five or 10 years. When you remove the old battery from the NES game, you will lose all your existing saved game data. But chances are that if you're doing this hack, your cartridge is unable to save games or keeps losing your games anyway.

Let's get started!

Preparing for the Hack

For this hack, you'll need two parts:

- A cartridge whose battery you want to replace
- A 3V lithium, CR2032-style coin-cell PCB mount battery (Digi-Key #P196-ND)

This battery will replace the old one within your game cartridge. Be sure to obtain the correct battery type—a regular CR2032 without the PCB mount leads (such as from Radio Shack and common drugstores) will not work, because you will not be able to easily solder it to the circuit board.

The tools required for this hack are:

- A soldering iron
- Solder sucker or solder braid
- A jeweler's size flathead screwdriver or NES 3.8mm security bit (depending on your cartridge type; see the “Opening an NES Game Cartridge” section for details)

Performing the Hack

Perform the following:

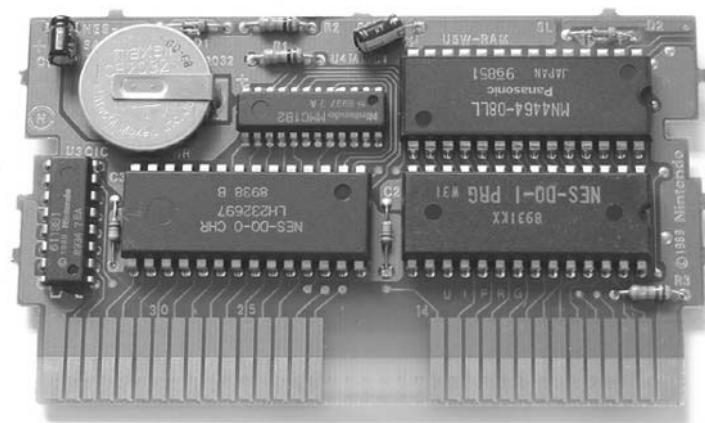
1. First, open your NES game cartridge as described in the “Opening an NES Game Cartridge” section earlier in this chapter. With the cartridge opened, remove the circuit board and flip it upright so that you can see the components. Your board should resemble the one shown in Figure 7.39.

NOTE



Many different types of circuit boards are used in NES games. Different games use different boards, depending on their program size and functionality, so your particular game might not resemble the image exactly. The important thing to look for is the round, quarter-sized battery. The battery that we will replace is usually located in the upper-left corner of the board, although the location may vary depending on the game. If your cartridge has one, you can proceed with this hack. For more information on the different types of game cartridge circuit board types, see the “Creating an EPROM Cartridge for Homebrew Game Development” section later in this chapter.

Figure 7.39 NES Game Cartridge with Battery-Backup Feature



WARNING: HARDWARE HARM

When you remove the old battery from the NES game, you will lose all your saved games and data. So, don't perform this hack unless you are unable to save games or if your game data mysteriously disappears.

2. With access to the circuit board, first unsolder and remove the old battery from the board (see Figure 7.40). Be careful not to apply too much heat to the battery leads, because excessive heat can cause the battery to explode. You might want to simply clip the leads of the battery first to remove the actual battery. Then all you have to do is remove the remaining battery leads from the holes.

WARNING: HARDWARE HARM

Do not heat the battery connections too much, because excessive heat could cause the battery to explode!

Also be careful when removing the battery from the board—you don't want to pull up the solder pads from the circuit board. Try to remove as much solder as possible from the joints on the bottom of the board, and then carefully pull the battery up from the board while heating the connections.

3. When the old battery is removed, your board should resemble the one shown in Figure 7.41.

Figure 7.40 Desolder the Battery

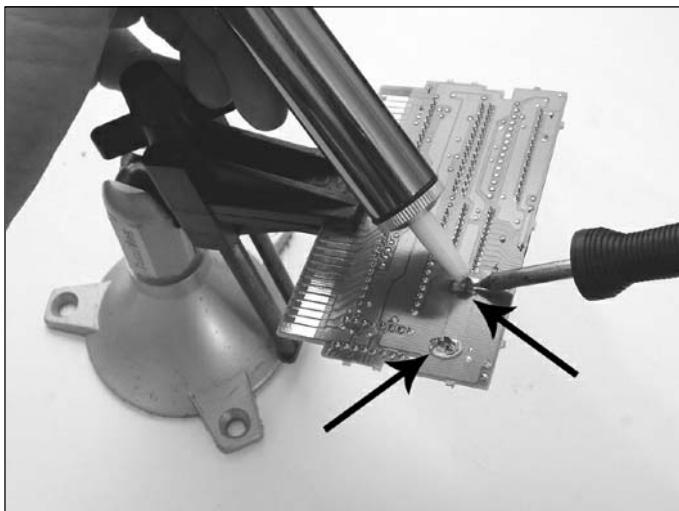
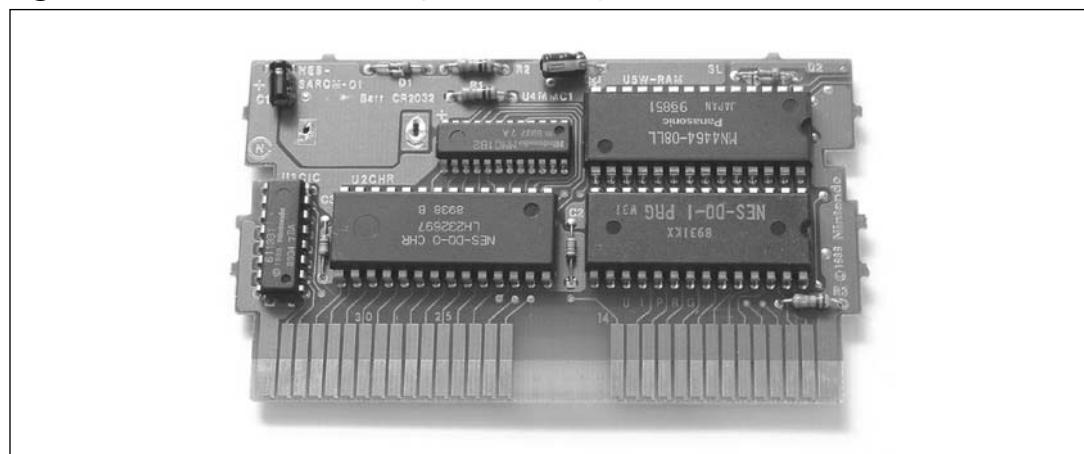
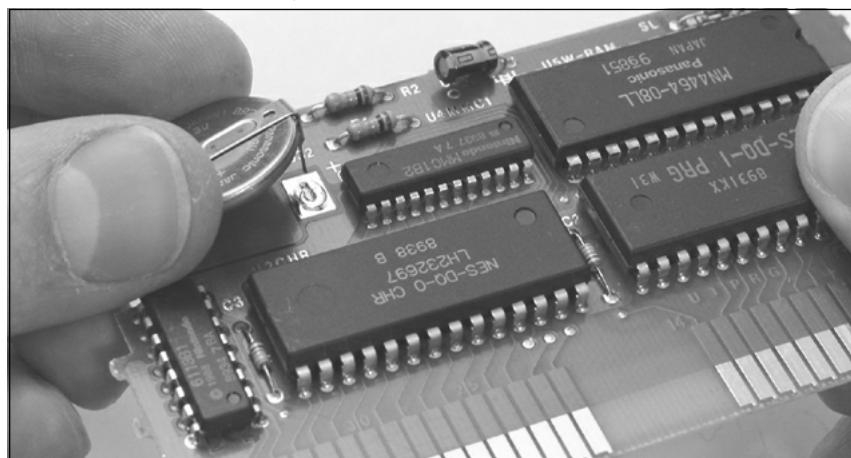


Figure 7.41 NES Game Cartridge with Battery Removed



4. Next, insert the new battery into the location of the old battery (see Figure 7.42). Be sure that the lead marked + (or positive) on the battery is inserted into the hole marked + on the circuit board.

Figure 7.42 Replace the Battery



5. Now solder the new battery onto the board.
6. To complete the reassembly, follow the steps as described in the previous “Opening an NES Game Cartridge” section. Once you reassemble your cartridge, the hack is complete and you can enjoy many more years of saving games in your favorite NES cartridges.

Creating an EPROM Cartridge for Homebrew Game Development

When playing videogames, nothing beats playing them on the actual hardware. In this day and age of PC-based emulators and videogame ROM images posted all over the Internet, people rarely return to the old days of sitting on the couch, controller in hand, staring slack-jawed at the television while they play. When developing a homebrew game for the NES (or any other platform), it is essential to test your program on actual hardware before releasing it. This is because the emulators, as perfect as some of them are, do not usually have the quirks and exact functionality of the true hardware.

The Erasable Programmable Read-Only Memory (EPROM) is not a permanently programmed device; rather, it can be erased with ultraviolet (UV) light and then reprogrammed when changes are required. This is useful for homebrew developers and programmers, who can program their game image into EPROMs, test the game on the NES console, erase the EPROMs, make the necessary changes to the code, and repeat the cycle until the game is complete.

This hack will guide you through the process of creating an EPROM-based cartridge that you can load games onto and use in an actual NES console. The hack sacrifices a standard Nintendo NES game cartridge and turns it into a development cartridge.

Preparing for the Hack

For this hack, you'll need the following parts:

- A Nintendo NES game cartridge (must be Mapper 0 style such as 10 Yard Fight, Arkanoid, Donkey Kong, Donkey Kong Jr., Duck Hunt, Excitebike, Ice Hockey, Slalom, or Super Mario Brothers; see “Under the Hood: How the Hack Works” at the end of this hack for more details)
- Two 28-pin DIP sockets (Digi-Key #AE8928-ND)
- An SPDT toggle or slide switch (Radio Shack #275-613 or Radio Shack #275-625)
- Three 6-inch lengths of 24AWG solid wire
- A 2764 8KB, 100-250nS EPROM (for the CHR-ROM, programmed with the desired binary image)
- A 27128 16KB or 27256 32KB, 100-250nS EPROM (for the PRG-ROM, programmed with the desired binary image)

The tools required for this hack are:

- A soldering iron
- Solder sucker or solder braid
- A jeweler's size flathead screwdriver or NES 3.8mm security bit (depending on your cartridge type; see the “Opening an NES Game Cartridge” section for details)

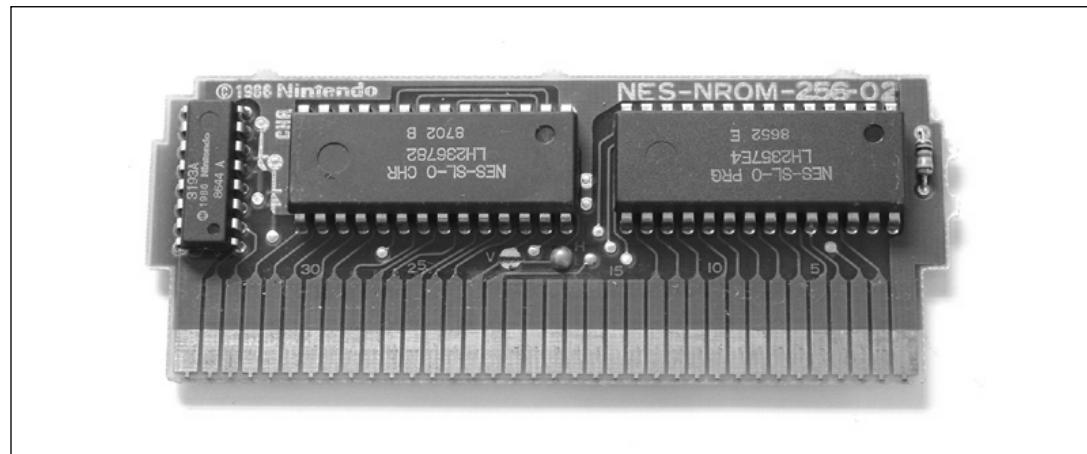
- A device programmer (to program the CHR-ROM and PRG-ROM binary images into the EPROMs)

Performing the Hack

Perform the following:

1. First, open your NES game cartridge as described in the “Opening an NES Game Cartridge” section earlier in this chapter. With the cartridge opened, remove the circuit board and flip it upright so that you can see the components. Your board should resemble the one shown in Figure 7.43.

Figure 7.43 NES Game Cartridge, Mapper 0 Style

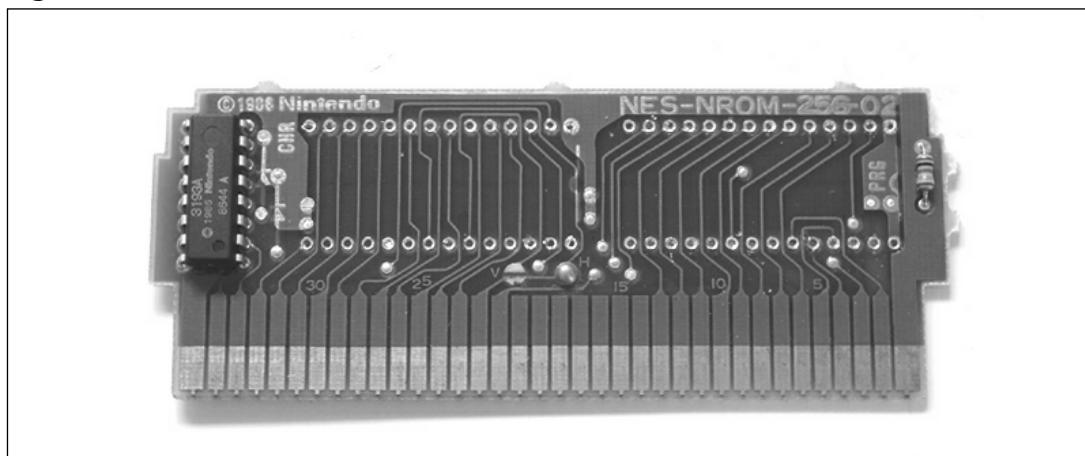


2. The next step is to desolder and remove the two original ROM devices from the game cartridge circuit board. When the components are removed, your board should resemble the one shown in Figure 7.44.

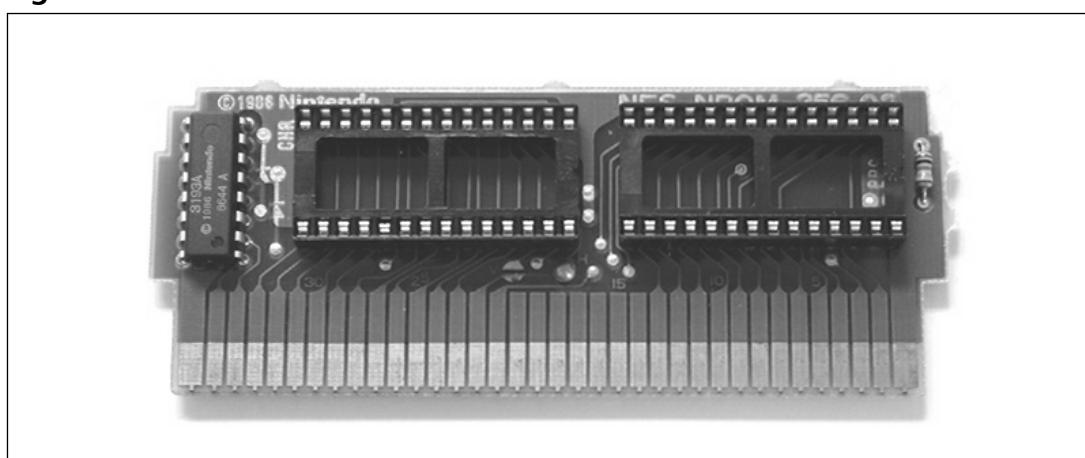
WARNING: HARDWARE HARM



Be careful when removing the ROM devices from the board—you don’t want to pull up the solder pads from the circuit board. Try to remove as much solder as possible from the joints on the bottom of the board, and then carefully pull the ROMs up from the board while heating the connections. If you are having difficulty removing the devices, use wire snips to cut the leads from the device packaging and then individually remove each pin from the board.

Figure 7.44 ROM Devices Removed from the Board

3. Now insert the 28-pin DIP sockets into the locations previously used by the ROM devices. When inserting the sockets, be sure to orient them with the notch of the socket aligning with the notch drawn on the circuit board (both should be facing right, as shown in Figure 7.45).
4. When the sockets are inserted correctly into the board, solder them into place.

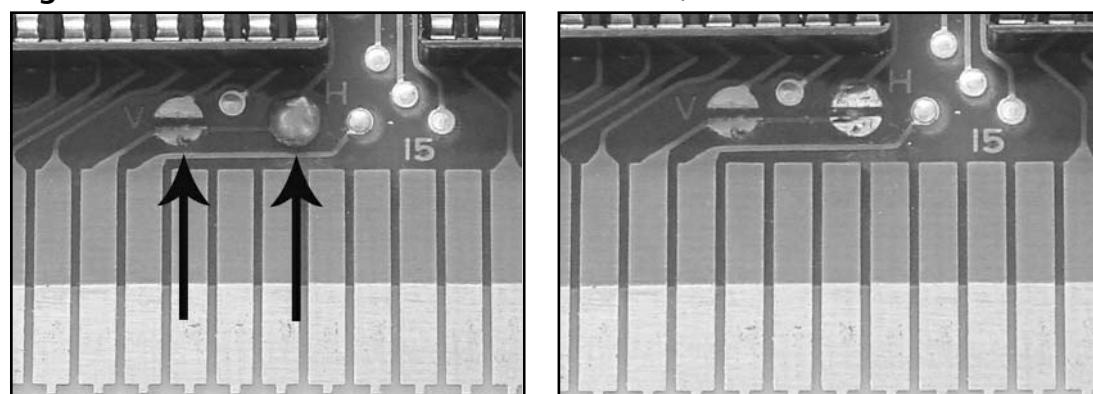
Figure 7.45 Sockets Added to the Board

5. At this point, if you want to test that you've correctly soldered the sockets to the board, you can insert both ROM devices into the sockets. Take care to ensure that they are placed in the same location and direction as they were originally, as in Figure 7.43. When you plug the board into your NES and power up the console, the original game should come up on

the screen, just as you would expect. Remove the ROM devices from the sockets to continue with the hack.

6. Next, look toward the bottom of the circuit board for sets of pads marked by an “H” and a “V.” Depending on the game you have chosen, either the H or V pad (but not both) will be connected with a ball of solder. Using the solder sucker, remove the ball of solder from the board. Figure 7.46 shows what your circuit board should look like before and after removing the solder from the pads.

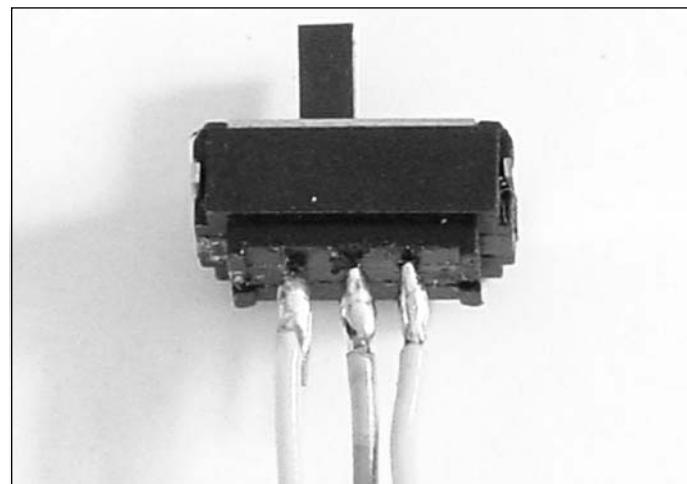
Figure 7.46 Remove Solder from the H and V Pads, Before and After



Now we need to install our switch. We will attach the switch to the top of the PCB and use three short lengths of solid 24AWG wire to make the connections.

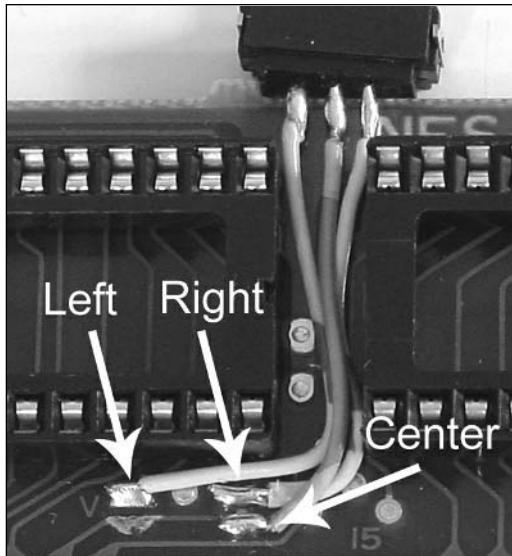
7. First, strip about 1/8 inch of shielding from the lengths of the 24AWG wire, and solder one end of each wire to the three leads of your SPDT switch (see Figure 7.47).

Figure 7.47 Switch Prepared with Wires



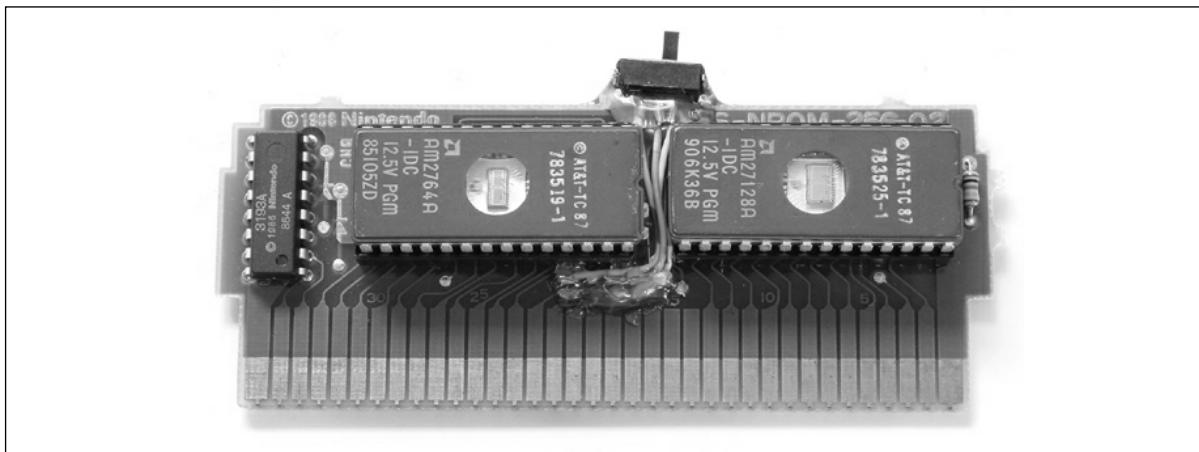
8. Next, solder the three leads of the switch to the H and V pads, as denoted in Figure 7.48. Take care that you are connecting the wires to the correct pads, since this is a critical step. The left lead from your switch should connect to the top half of the V pad. The right lead from your switch should connect to the top half of the H pad. The center lead from your switch should connect to the bottom half of the H pad.

Figure 7.48 Switch Soldered to the H and V Pads



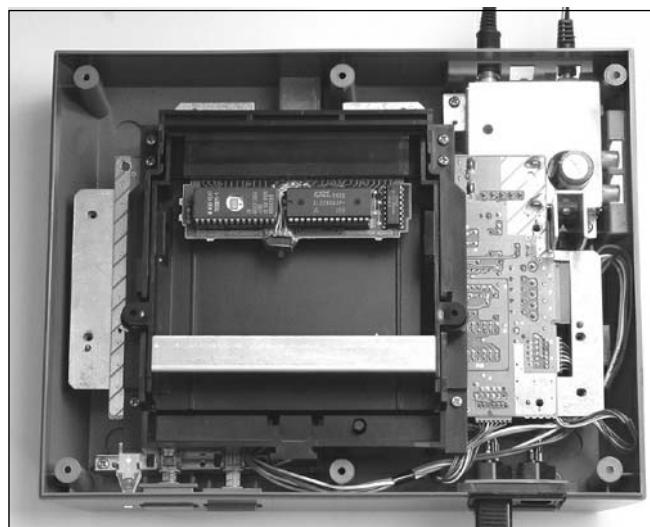
9. With the switch soldered into place, the last thing we need to do is fasten the switch to the top of the circuit board with hot glue. In addition, you might want to use hot glue over the new solder connections on the circuit board to protect them and act as a strain relief.
10. For the board to function properly, you need two EPROM or Flash memory devices: one 8KB device programmed with the CHR-ROM image and one 16KB or 32KB device programmed with the PRG-ROM image. Use your device programmer to program your desired binary game images into the EPROMs and insert them into the cartridge. Refer to the “Homebrew Development” section of this chapter for information on obtaining or creating homebrew NES games. Your EPROM cartridge should resemble the one shown in Figure 7.49.

The hack is now complete. Enjoy the thrill of playing your homebrew games on real hardware! If your game doesn't load or the graphics are displayed improperly, try flipping the switch and turning the console back on.

Figure 7.49 Completed NES EPROM Cartridge

Be aware that once you hack together your EPROM development cartridge, it won't fit back into the NES cartridge case and won't fit into the original NES Toaster console, since you have to insert the cartridge deeply into the system. The circuit board (without the cartridge housing) should still be able to plug into the later U.S. release of the NES, the Top Loader.

The best way to use your cartridge is to open the case of the NES Toaster and remove the RF shield, as shown in Figure 7.7 and as explained in the “Opening the NES Console” section of this chapter. This will give you access to the cartridge connector, and you will have enough vertical clearance for your new EPROM development cartridge. Ensure that the cartridge tray is pressed down until you hear the click so that the NES thinks that a full cartridge has been inserted properly (see Figure 7.50).

Figure 7.50 EPROM Cartridge Inserted into the NES Console

Under the Hood: How the Hack Works

Nintendo game cartridges use a method known as *bankswitching* to increase the amount of space and storage configuration of game and program data and to control how data is accessed in a particular cartridge. Bankswitching is a method used to overcome a memory size limitation and allows a larger-size memory device to be used with a cartridge, thus providing more data storage for the system. Bankswitching requires specially designed logic circuitry on the cartridge to handle the specific schemes. Nintendo refers to its various schemes as *mappers*.

The Mapper 0 configuration (also referred to as *NROM*) is commonly used for NES homebrew games because it is the simplest form to develop for and only simple modifications are necessary to create an EPROM cartridge. The NES cartridge has two separate memory areas: CHR-ROM, used for the storage of character information and data, and PRG-ROM, used for storage of the game program code. In a Mapper 0 cartridge, the CHR-ROM is usually 8KB and the PRG-ROM is usually 16KB or 32KB.

The primary reason for using a Mapper 0 cartridge as an EPROM cartridge is that the pinout of the original ROMs on the game cartridge circuit board are pin-for-pin compatible with standard EPROM memory devices (which can be programmed and reprogrammed at will). As such, no major modification to the cartridge is necessary to make our EPROM development cartridge.

The switch we added allows us to switch between the H (horizontal) and V (vertical) pads, which are used to select the ROM mirror configuration that the game uses. Mirroring, in this case, specifies how the graphics are stored in the CHR-ROM and varies depending on the game. The switch lets you use a single development cartridge for both types of mirroring scheme.

Many different types of cartridges and mapper schemes exist. For an exhaustive list of NES titles, mapper types, ROM sizes, and H/V mirroring configurations, refer to the “Bigass NES Mapper List” located at <http://tuxnes.sourceforge.net/nesmapper.txt>.

For an excellent, detailed description of the known mapper schemes, refer to the “Comprehensive NES Mapper Document” located at <http://tuxnes.sourceforge.net/mappers-0.80.txt> and Kevin Horton’s NES Mappers page located at <http://tripoint.org/kevtris/mappers/mappers.html>.

Many hacks are available on the Web for creating development cartridges from other mapper schemes. See the links in the “Other Hacks” section of this chapter for more information.

Homebrew Game Development

Although the Nintendo NES has been relegated to the category of classic videogame system, it still has a fierce following of loyal fans who enjoy the original games as well as continue to create new games for it. Even though hobbyists write these games in their spare time, usually just to experiment with the hardware or program a game that has never been done before, many of today’s homebrew games rival the quality of the games from “back in the day.”

The core NES hardware consists of a Motorola 6502 8-bit processor running at a 1.79MHz, 2KB of system RAM, and a custom Picture Processing Unit (PPU). The screen resolution is 256 x 240 pixels, and the console can draw 24 colors on the screen at one time, out of an available 53 colors (48 colors and 5 grays). Although the system is limited in resources compared to today’s game consoles and personal computers, the thrill and challenge of programming for the NES draw people to it.

Homebrew authors for the NES today have many advantages over programmers who were writing games for the NES in the mid-1980s. Thanks to the Internet, a vast warehouse of information on how to program the NES is available online, including source code, tutorials, development tools, emulators, and custom hardware. The following links will serve as a good starting point for your adventures into NES homebrew programming:

- **Game Development for the 8-Bit NES, Course #98-026, Carnegie Mellon University, <http://bobrost.com/nes/index.php>** Currently the most talked-about programming resource for the NES. Bob Rost, a teacher and game developer, teaches the course at Carnegie Mellon University. The course description states that the “class will teach students how to create videogames for the 8-bit Nintendo Entertainment System. Students in the class will work together in small development teams (or alone, if preferred) to create games, demos, and development tools. One game, demo, or development tool will be created for mid-semester, and one game will be created for the end of the semester.” The lecture notes and resources are excellent and provide technical details into the NES hardware as well as instruction and tips into writing games for the system. A must-see for any aspiring NES homebrew programmer.
- **NesDev, <http://nesdev.parodius.com>** An excellent and comprehensive site on all things technical for the NES. The site is the primary depot for NES development information and resources. The site is updated often with documentation, homebrew games and programs, development tools, and links. Everything you would ever want to know about the technical aspects of the NES systems and associated hardware and peripherals can be found here. Another must-see for any aspiring NES homebrew programmer.
- **NES Technical/Emulation/Development FAQ, www.zyx.com/chrisc/NESTechFAQ.html** Maintained by Chris Covell, this exhaustive document contains just about everything you would want to know about NES hardware, emulation, and programming.
- **Jnes NES Emulator, www.jabosoft.com/jnes** Jnes is one of the most accurate Win32-based NES emulators. There are dozens of other available emulators for the NES system, a list of which can be found here: www.zophar.net/nes.html.
- **tniNES: An NES ROM Editing Utility, www.patriekl.dds.nl/tniNES.html** One of the many useful NES development tools, tniNES is used to edit and manipulate NES ROM images in iNES format (.NES files), used by emulators like iNES, NESticle, and fwNES. Using tniNES, you can split an .NES game image file into the separate .PRG and .CHR images required for use with an EPROM development cartridge.
- **#nesdev IRC (Internet Relay Chat) Channel** Located on the EFnet server (irc.efnet.org), the #nesdev channel has been in existence for many years and is dedicated to discussing programming and development for the NES.

Other Hacks

The hacks presented in this chapter are only the beginning of what you can do with your NES. We've listed some additional hacks here:

- **Stereo Audio Output Modification,** www.zyx.com/chrisc/nesstereo.html This simple hack describes how to modify your NES console to provide stereo audio output (instead of the regular mono signal). It's unknown what original NES games (if any) would sound better with stereo audio, but it would be something that new homebrew games could take advantage of.
- **Composite Audio/Video Output for an NES 2 (Top Loader),** www.gamesx.com/rgbadd/nes2avmod.htm When Nintendo released the second U.S. version of the NES in the mid-1990s, the company removed the composite video output from the back of the unit, much to the chagrin of many gamers. Composite video provides an improved picture signal compared to the standard RF output included on the system. This composite A/V hack can be done in a few minutes with a few dollars' worth of components.
- **The “Bratwurst Toaster” Deck Modification,** www.angelfire.com/apes/mad-meat/toaster1.html This extreme hack fixes Nintendo's notoriously poor 72-pin cartridge connector used in the NES Toaster by removing it completely, instead of simply replacing the connector (as described in the “Replacing the 72-Pin Cartridge Connector” section earlier in this chapter).
- **Creating EPROM Cartridges for Specific Mappers** Nintendo game cartridges use a method known as *bankswitching* to increase the amount of space and storage configuration of game and program data and to control how data is accessed in a particular cartridge. Bankswitching is a method used to overcome a memory size limitation and allows a larger-size memory device to be used with a cartridge, thus providing more data storage for the system. Bankswitching requires specially designed logic circuitry on the cartridge to handle the specific schemes. Nintendo refers to its various schemes as *mappers*. As discussed in the “Creating an EPROM Cartridge for Homebrew Game Development” section, there are different types of mapper schemes for various NES games. Creating EPROM development cartridges from original NES game circuit boards will vary depending on the mapper scheme; the following Web sites provides information on how to do so:

- **Mapper 0, 1, 2, 3, 4, and 7,**
<http://nesdev.parodius.com/NES%20EPROM%20Conversions.txt>
- **Mapper 2,** www.planetnintendo.com/thewarpzone/tech/UNROMdev.txt

NES Resources on the Web

There are a great number of resources on the Web for enthusiasts of the Nintendo NES game console, though some aren't updated on a regular basis. Whether you're looking for information about your favorite games, for ways to hack your system to give it capabilities never intended by the designers, or for forums to wax nostalgic with other Nintendo fans, you're sure to find it online somewhere. The following is a list of some of our favorite Nintendo NES-related sites:

- **NES World,** www.nesworld.com Arguably the most comprehensive NES resource on the Web, this site includes information on NES systems, games, and homebrew development, featuring a large collection of articles, interviews, technical data, and other written material. The site also has a section dedicated to magazine and TV advertisements, posters and pamphlets, books and videos, scans of box art, and screenshots of games.
- **Nintendo Land,** www.nintendoland.com A site dedicated to preserving information on Nintendo's classic game systems. Contains an art gallery, fun facts and history of Nintendo, an active message board, online games, and many articles and reviews. The site also hosts three subsites: Mario Mania, Zelda: The Grand Adventures, and Planet Zebes, honoring Nintendo's popular videogame heroes.
- **The Warp Zone,** www.planetnintendo.com/thewarpzone A tribute site to the NES and Famicom systems. Provides game tips, information and pictures of rare Nintendo accessories, articles, interviews, and a wealth of miscellaneous other documentation.
- **NES Player,** www.nesplayer.com Another comprehensive NES resource that includes information and background of the hardware, games, and companies involved in Nintendo development. The site also features news, reviews, editorials, forums, projects, and other special features.
- **NEShq.com,** www.neshq.com Contains a wide variety on information about the NES, including general and technical information on emulation, games, hardware, and projects. The original intent of the site was to provide a standardized location for all the disparate NES information floating around the Internet.
- **JunKmachine: Where Old School Is Still Cool,** www.junkmachine.com Dedicated to NES case modifications. Provides an active forum to discuss and show off your NES hacks.
- **smackdown GT,** <http://smackdown.myrmid.com/smackdown> An off-the-wall, humorous site containing NES-fueled information, articles, features, and other random commentary.

- **NES City, www.nescity.com** An NES-related fan site containing editorials, graphics, icons, and interesting stories.
- **The New NES Forums, <http://s8.invisionfree.com/nesforums>** An active forum for everything NES.
- **RetroNES, www.retrones.com (Spanish)** An excellent collection of NES information, including articles, tutorials, details on accessories, pictures of Nintendo merchandise, and much more.

Atari 2600

Topics in this Chapter:

- **Introduction**
- **Atari 2600 Left-Handed Joystick Modification**
- **Repair Your Atari 2600 Joysticks**
- **Revitalize Your Atari 2600 Paddles**
- **Use an NES Control Pad with Your 2600**
- **Atari 2600 S-Video/Audio Mod**
- **Atari 2600 Stereo Audio Output**
- **Homebrew Game Development**
- **Atari 2600 Resources on the Web**

Introduction

When Atari introduced the Video Computer System (VCS) back in 1977, nobody, not even Atari, knew it would ultimately become a wild success and be the catalyst that would spawn the multibillion-dollar gaming industry we know today. The VCS (see Figure 8.1), later renamed the 2600, was part of the first generation of videogame systems that weren't hardwired to play a certain set of games. Instead, they used removable and replaceable videogame cartridges. This flexibility helped propel the VCS to the top of the sales charts, where it sat as king for many years until the videogame market crashed in 1984.

Figure 8.1 A First-Generation Atari Video Computer System



The Atari VCS initially shipped with a pair of joystick controllers, a pair of paddle controllers, and the two-player game *Combat*. Atari released additional controllers for the system, such as the driving controllers (used by only a single game—*Indy 500*), keypad controllers, and even a Trak-Ball controller. The initial library of 2600 titles was small (nine games were offered), but the 2600 catalog would eventually encompass hundreds of titles produced by a wide range of companies.

One important first for the young videogame industry was the creation of Activision in 1980, formed by several Atari employees who were dissatisfied with the way they were being treated at Atari. Activision was the first third-party company to produce videogames for the Atari 2600, a move that Atari tried to quash in the courts, with no success. Activision's high-quality games were well received by the public, and Activision grossed \$70 million in its first year.

The Atari 2600 lived a long life, if somewhat bumpy in later years. In 1986 Atari released the 2600 Jr., a 2600 repackaged into a smaller case with a \$50 price tag and a fresh marketing campaign. Atari continued to push the 2600 in the United States until 1989, at which time production for the console was finally ceased. With a run lasting over 10 years and millions of consoles sold, the Atari

2600 is one of the most successful game consoles of all time, with a place in history as the machine that started the home videogame craze.

Hacks in This Chapter

Although the Atari 2600 is now nearly 30 years old, it has a large fan base that keeps the system alive and thriving in the 21st century. Hundreds, if not thousands, of Web sites are devoted to the game system that formed the basis of the videogame industry today. A large community of programmers and hobbyists still create new games for the 2600, with access to resources the original programmers could only dream of.

In this chapter we'll cover how to do the following hacks for the Atari 2600, which only scratch the surface of what can be done:

- Create an Atari 2600 left-handed joystick
- Repair your Atari 2600 joysticks
- Revitalize your Atari 2600 paddles
- Use an NES control pad with your 2600
- Add stereo output to your 2600
- Complete an Atari 2600 S-Video/composite modification

If the hacks detailed here whet your appetite, we've put together a list of additional hacks that you can pursue by following the links at the end of this chapter. We've also put together a section on Atari 2600 homebrew development for those of you who would like to get your feet wet writing new games for the Atari 2600.

Let's get started!

Atari 2600 Left-Handed Joystick Modification

The Atari CX-40 joystick that shipped with the Atari 2600 is probably the single most recognized controller in videogame history. However, in the late 1970s, ergonomic controller design was not a concern of hardware designers, and in the case of the venerable CX-40, left-handed players were ignored. The CX-40, pictured in Figure 8.2, was designed to be held with your left hand, leaving your right hand to operate the joystick. The single joystick button is then triggered with your left thumb. Unfortunately, this arrangement is counter-intuitive for those who are left-handed.

Figure 8.2 Atari CX-40 Joystick

The goal of this hack is to transform a stock CX-40 joystick into one that can more easily be enjoyed by left-handed players. We will do this by modifying the joystick so that it can be rotated 90 degrees clockwise. This will place the fire button in the upper-right corner, where it can be operated by your right thumb, leaving your left hand free to manipulate the joystick.



Preparing for the Hack

This is a fairly simple hack that requires only:

- An Atari CX-40 joystick
- A Phillips head screwdriver (used to open the joystick)
- A pair of needlenose pliers (used to disconnect wires from the PCB)

You should select a joystick that is in good operating condition. If your joystick is not working well before the hack, it's not likely to serve you well afterward. This hack can be easily reversed if you decide you don't like the rotated nature of the joystick.

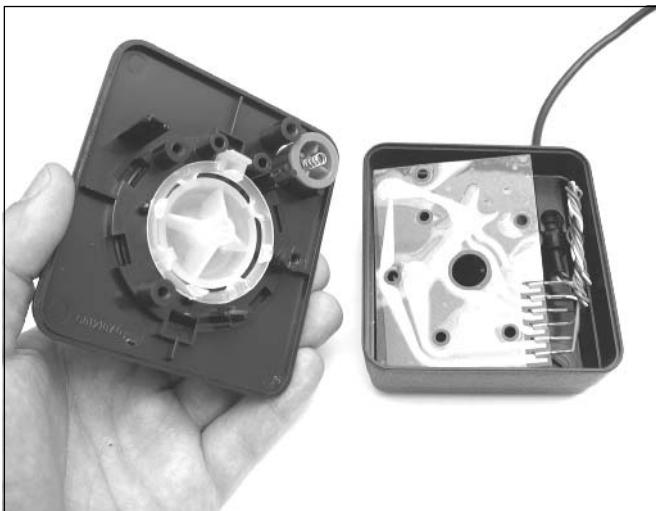
Performing the Hack

Perform the following steps:

1. Flip the joystick over and use a Phillips head screwdriver to remove the four screws, as shown in Figure 8.3.

Figure 8.3 Remove the Four Screws

2. With the joystick still upside down, carefully remove the base. There is a small spring beneath the orange fire button—be careful not to lose it! Pull the top half of the joystick away; this half will contain a clear plastic post (which gives the joystick its rigidity), the orange fire button, and the spring inside the button (see Figure 8.4). Put these components aside for now.

Figure 8.4 Taking the Joystick Apart

The joystick base contains a simple circuit board to which six wires are attached. These wires represent the four joystick directions, the fire button, and the ground.

- Pull the circuit board out from the base to make it easier to work on. Using your fingers or a pair of needlenose pliers, swap the connectors according to Table 8.1 and as shown in Figure 8.5.

WARNING: HARDWARE HARM

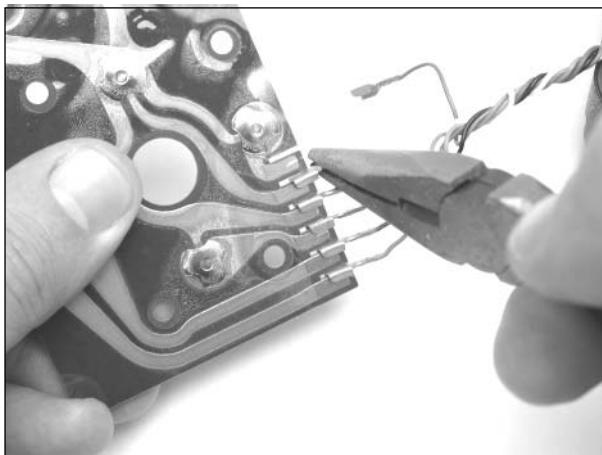


When pulling the connectors out, make sure you are pulling by the metal and *not* on the wire! If you pull on the wire, you could separate the wire from the connector. If this happens, you will have to solder the wire back onto the connector.

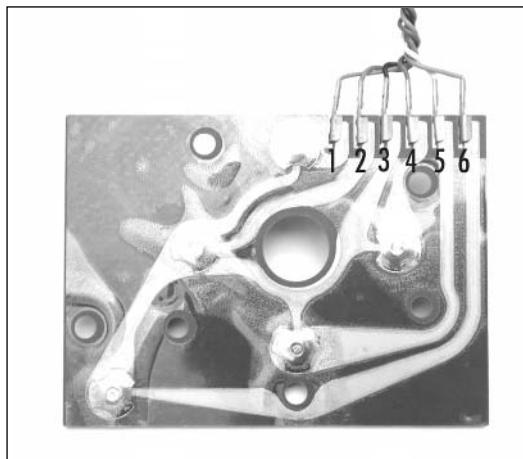
Table 8.1 Atari CX-40 Joystick Rewiring

Connector	Original Connection Color	New Color
1	Brown	Blue
2	White	Brown
3	Black	Black
4	Blue	Green
5	Green	White
6	Orange	Orange

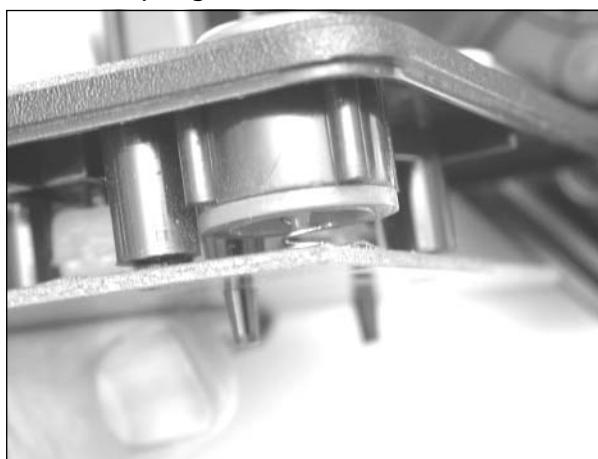
Figure 8.5 Swapping the Wires



- After completing the wire swap, your board should resemble the one shown in Figure 8.6.

Figure 8.6 Final Wire Positions

5. After double-checking your work, you can now reassemble the joystick. The easiest way to do this is to hold the top half of the joystick upside down and place the circuit board on top of it, aligning the holes of the PCB with the plastic posts of the case. Take care to ensure that the spring for the joystick button is properly sandwiched between the button and the board, as shown in Figure 8.7.

Figure 8.7 Joystick Button Spring

6. Place the joystick base on top of the other half of the joystick, making sure that the wires are bundled up along the side of the joystick and not underneath the circuit board. The two halves of the joystick should fit together easily; if they do not, check to make sure that the wires are not encroaching on the board or between the two halves. Insert the four screws into the base of the joystick and tighten them with the Phillips head screwdriver.

Now you can test your creation! If you decide you don't like the way the joystick feels, you can easily reverse this hack by returning the wires back to their original positions.

Repair Your Atari 2600 Joysticks

The Atari CX-40 joystick that shipped with every 2600 system is a fairly simple device, but it is susceptible to failure after prolonged use. Fortunately, parts for the CX-40 joystick are easily obtained and repairing a broken joystick takes only a few minutes.

There are two common failures for the Atari CX-40 joystick. Inside the rubber sheath of the joystick is a stiff, plastic handle. At the base of this handle is a ring that contains four protrusions that come into contact with the metal buttons on the circuit board below it. This plastic ring can break, rendering one or more joystick directions unresponsive. The second common failure is of the circuit board itself. The circuit board contains five metal buttons, four of which are depressed by the joystick (one for each direction—left, right, up, and down), and a fifth, which is triggered by pressing the orange fire button. These buttons can also fail over time, again rendering the joystick inoperable. This hack explains how to replace these two parts in the CX-40 joystick.

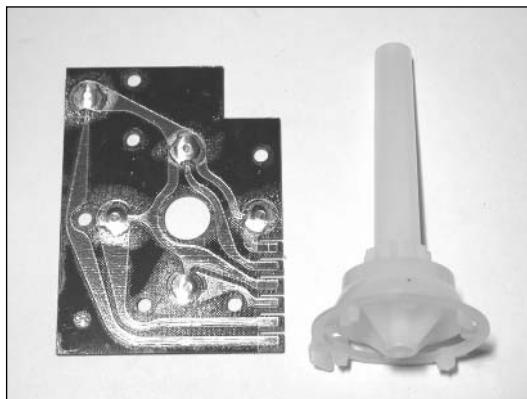
Preparing for the Hack

To perform this hack, you'll need:

- Replacement CX-40 joystick parts
- A Phillips head screwdriver
- A pair of needlenose pliers (optional)

Before getting started, you'll need to purchase replacement CX-40 joystick parts. Best Electronics (www.best-electronics-ca.com) is a good source for the plastic joystick handle and the printed circuit board, and the company sells a kit combining both of these parts in one package (see Figure 8.8).

Figure 8.8 CX-40 Replacement Parts



If you choose to purchase an optional improved version of the Atari CX-40 joystick handle, you can also find this at Best Electronics. This improved handle has a reinforced ring with stress-relieving slots where the ring typically fails. Figure 8.9 shows the improved handle on the left and the original handle on the right.

Figure 8.9 Atari CX-40 Joystick Handles—Left: Improved, Right: Original

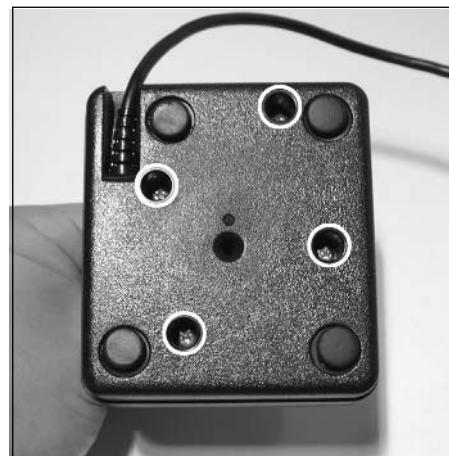


Performing the Hack

Perform the following:

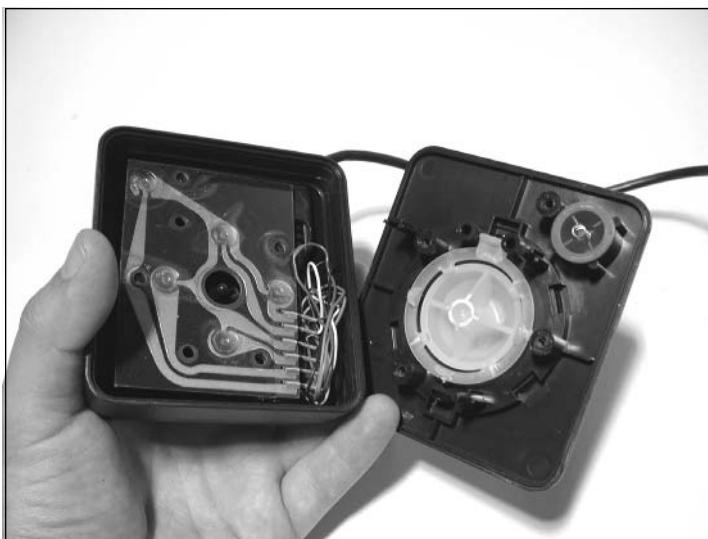
1. Flip the joystick over and use a Phillips head screwdriver to remove the four screws, as shown in Figure 8.10. Place the screws aside for reassembly later.

Figure 8.10 Removing Four Screws



2. With the joystick upside down, carefully lift up the base, separating the two halves of the joystick. While removing the base, be careful not to lose the small spring beneath the orange fire button, or your fire button will no longer work and your joystick will be useless! (See Figure 8.11.)

Figure 8.11 Taking the Joystick Apart



3. Once you have the two halves of the joystick separated, you can inspect the plastic handle to see if the ring contains any breaks. If it does not, you don't need to replace it, but you can if you want. To do so, you first need to remove the original plastic handle from the rubber sleeve—simply pull it out. It might need some coercing if the plastic handle resists and is adhering to the rubber. You can wiggle it a bit while pulling on it, and it will eventually come free, as shown in Figure 8.12.

Figure 8.12 Removing the Plastic Handle



- Once you have the plastic handle removed, you can insert the replacement handle. To do so, orient the rectangular tab so that it is at the 12 o'clock position relative to the top of the joystick. Push the handle all the way into the rubber housing until it will slide no further. The replacement handle should resemble Figure 8.13. As you can see in the figure, we used the improved joystick handle with the stress-relief slots.

Figure 8.13 Handle Replaced



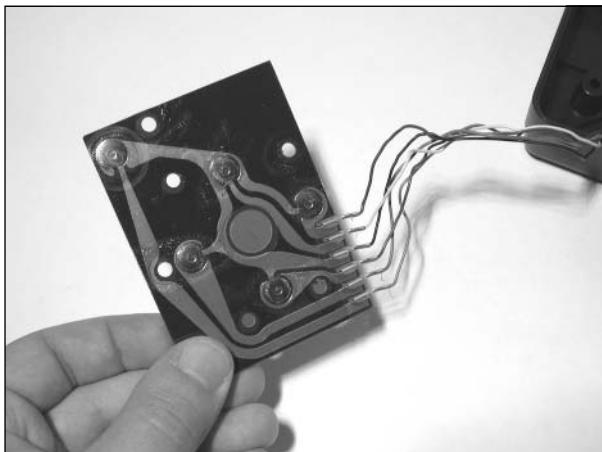
Now we'll move onto the second half of the repair—that of the printed circuit board. The joystick base contains a simple circuit board to which six wires are attached. These wires represent the four joystick directions, the fire button, and ground. Figure 8.14 shows the circuit board sitting in the base of the joystick, with wires attached.

Figure 8.14 Joystick Base



5. First, remove the circuit board by lifting it out and moving the base aside, as shown in Figure 8.15.

Figure 8.15 Removing the Circuit Board



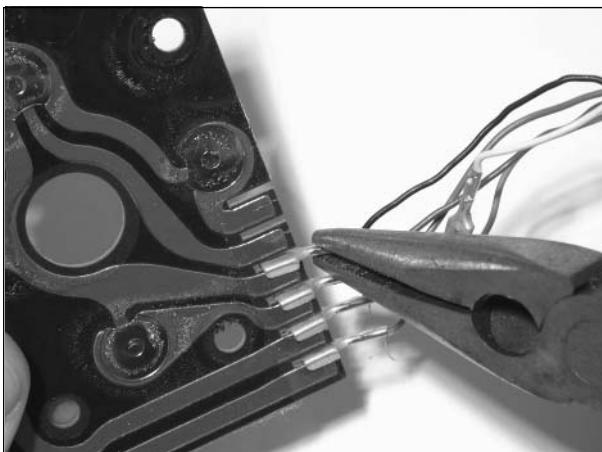
6. Now that you have the circuit board separated from the plastic joystick base, it's time to disconnect the wires from the board. Using your fingers or a pair of needlenose pliers, disconnect the six wires, as shown in Figure 8.16.

WARNING: HARDWARE HARM



When pulling the connectors out, make sure you are pulling by the metal and *not* on the wire! If you pull on the wire, you could separate the wire from the connector. If this happens, you will have to solder the wire back onto the connector.

Figure 8.16 Disconnecting the Wires



7. With the wires disconnected from the circuit board, you can discard the old circuit board. Take the new circuit board and attach the wires to the same positions they were attached to on the old board, as shown in Figure 8.17. The proper location of each wire is shown in Table 8.2.

Figure 8.17 Connect Wires to New Board

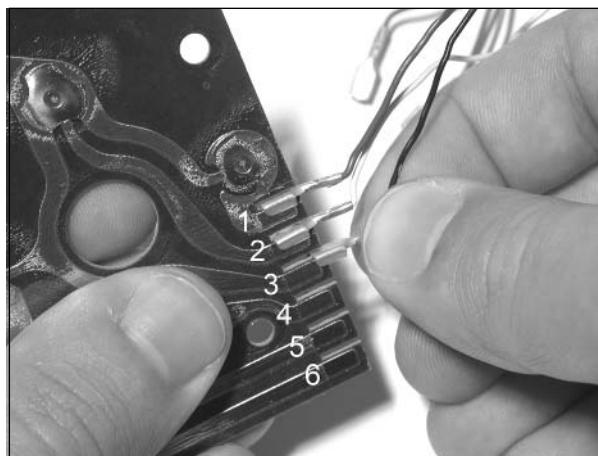
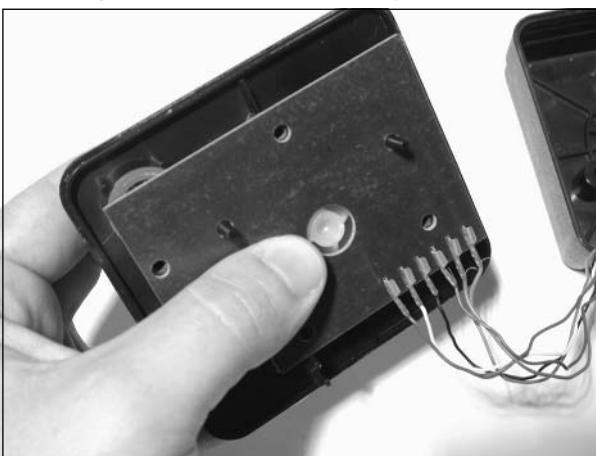


Table 8.2 Atari CX-40 Joystick Rewiring

Connector	Wire Color
1	Brown
2	White
3	Black
4	Blue
5	Green
6	Orange

8. With the wires connected to the new circuit board, you can now reassemble the joystick. The easiest way to do this is to hold the top half of the joystick upside down and place the circuit board on top of it, aligning the holes of the board with the plastic posts of the case, as shown in Figure 8.18. Take care to ensure that the spring for the joystick button is properly sandwiched between the orange fire button and the board.

Figure 8.18 Reassembling the Top Half of the Joystick

9. Place the joystick base on top of the other half of the joystick, as shown in Figure 8.19. Make sure that the wires connected to the circuit board are bundled up along the side of the joystick and not underneath the circuit board. The two halves of the joystick should fit together easily; if they do not, check to make sure the wires are not encroaching on the board or between the two halves.

Figure 8.19 Putting the Two Halves Together

10. Insert the four screws into the base of the joystick and tighten them with the Phillips head screwdriver, as shown in Figure 8.20.

Figure 8.20 Screwing the Case Together

With the repaired joystick fully assembled, go ahead and try it out! Your joystick should operate like new and with care will last you many years.

Revitalize Your Atari 2600 Paddles

For several years, Atari included a pair of CX-30 paddle controllers with every Atari 2600 system. Due to their analog nature, these paddle controllers (see Figure 8.21) bring a fairly unique control element to the games that use them. Based on the paddle controllers included with many older Pong-style systems, the Atari 2600 paddle controllers include a fire button and are comfortable to hold in your hand. They provide a precise means of controlling movement along one axis on screen in an analog fashion, as opposed to the digital nature of the CX-40 joystick. A player's success in paddle-based games depends greatly on mastery of the paddle controller.

Figure 8.21 Atari CX-30 Paddle Controllers

Unfortunately, the Atari 2600 paddle controllers deteriorate in precision over time, resulting in “jitter” that makes paddle games difficult to play. This jitter is caused by the buildup of dirt and grime in the potentiometer (also known as a *variable resistor*) within the paddle. Thankfully, this is easily remedied with simple disassembly of the paddle and a can of contact cleaner. Once you follow the next procedure, you should have a pair of paddles that feel as responsive and accurate as the day they left Atari’s factory.



Preparing for the Hack

For this hack, you’ll need:

- A can of contact cleaner, such as the Techspray Contact Cleaner shown in Figure 8.22. (Contact cleaner can be purchased at many electronics stores, including Radio Shack, Fry’s Electronics, and Digi-Key.)
- A Phillips head screwdriver (used to open the paddle controllers)
- A pair of needlenose pliers (used to unscrew the potentiometer nut)

Performing the Hack

Paddle controllers come in pairs of two controllers attached to a single connector. We’ll describe how to clean a single paddle controller, and you can repeat the process for the second controller.

Perform the following:

1. First, pull the plastic knob off the paddle by pulling it straight out (see Figure 8.23). It should pop off with little resistance. Put this knob aside for now.

Figure 8.22 Contact Cleaner



Figure 8.23 Removing the Plastic Knob



2. As shown in Figure 8.24, use the needlenose pliers to unscrew the nut sitting under the plastic knob you just removed. This nut keeps the potentiometer inside the paddle controller in place.

Figure 8.24 Unscrewing the Potentiometer Nut



3. Now flip the paddle controller over and remove the two screws from the bottom of the paddle controller, as shown in Figure 8.25.

Figure 8.25 Removing the Screws



4. With the screws removed, you can now pull the two halves of the paddle controller apart (see Figure 8.26). Be careful not to disturb the orange button assembly, which sits in place within the top half of the unit. If the button assembly pops out, you can reassemble it, but

it's easier if it is not removed during this procedure. Place the bottom half of the paddle enclosure aside for now.

Figure 8.26 Successful Disassembly of the Paddle



5. Sitting in the recess in the top half of the paddle controller is the potentiometer, the heart of the paddle controller. Remove the potentiometer from the plastic case, as shown in Figure 8.27.

Figure 8.27 Removing the Potentiometer

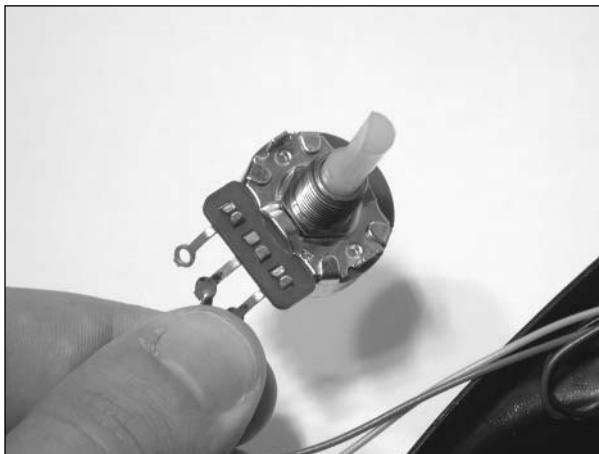
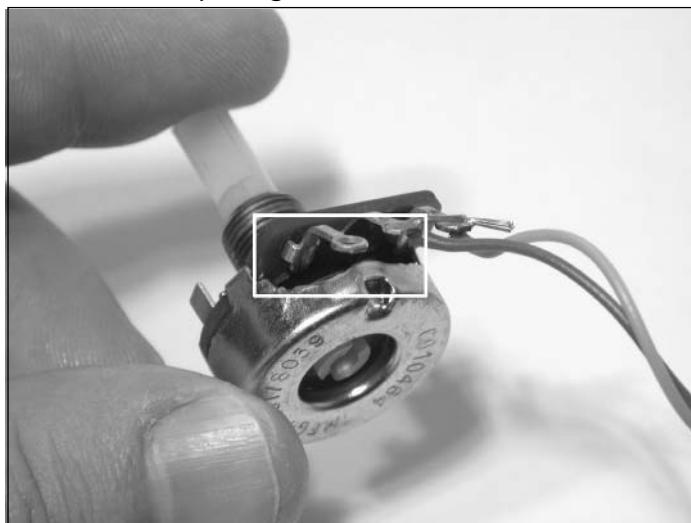
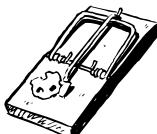


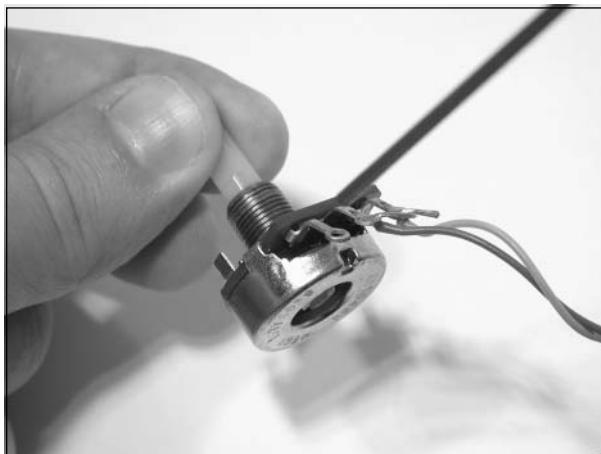
Figure 8.28 highlights an opening in the potentiometer where we'll be spraying the contact cleaner. Though it's not visible in the figure, if you look carefully at your potentiometer, you'll be able to see a metal contact that touches a disc that rotates around the inside of the potentiometer.

Figure 8.28 Potentiometer Opening

6. Using the can of contact cleaner, spray some solution into this opening, as shown in Figure 8.29. If the can of contact cleaner you purchased has a thin, plastic straw to help direct the flow of solution, attach it to the spray nozzle; doing so will make it easier to spray the cleaner directly into the potentiometer opening. After spraying the contact cleaner into the potentiometer, use your fingers to rotate the potentiometer through its full range of motion several times. Repeat this step two to three times.

WARNING: HARDWARE HARM

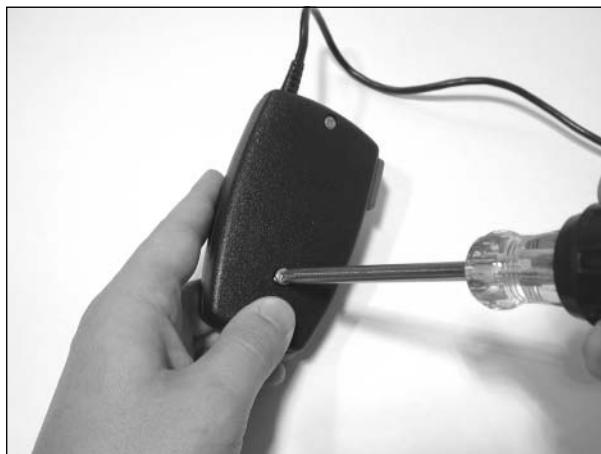
Take extra care not to spray any contact cleaner onto the plastic halves of the paddle controller or the orange plastic button. Doing so will cause the plastic to warp, and if that happens the pieces might not fit together properly.

Figure 8.29 Spraying the Contact Cleaner

7. After (and only after!) the contact cleaner has completely dried, place the potentiometer back into the paddle controller shell. Orient the contacts so that they are facing down relative to the controller, as shown in Figure 8.30. Place your index finger on the potentiometer to hold it in place, and thread the metal nut over the post of the potentiometer on the other side.

Figure 8.30 Replacing the Potentiometer

8. Next, carefully place the bottom half of the controller over the top half, making sure that no wires are pinched in between the two pieces. Insert and tighten the two screws that you removed earlier (see Figure 8.31).

Figure 8.31 Replacing the Bottom Half of the Paddle

9. After you've reassembled the two halves of the controller, you can now tighten the nut holding the potentiometer in place, as shown in Figure 8.32. It can be tightened most of the way with your fingers, but you'll want to use the needlenose pliers to tighten it fully.

Figure 8.32 Tightening the Nut

10. You can now reattach the plastic knob you removed in the first step, as shown in Figure 8.33.

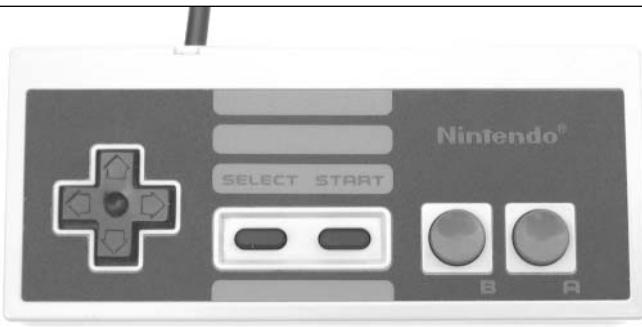
Figure 8.33 Reattaching the Knob

You have successfully cleaned your paddle controller! You can now repeat this process for the second paddle controller. When you're done, your paddle controllers should respond accurately and with no annoying jitter, making paddle-based games considerably more enjoyable.

Use an NES Control Pad with your 2600

In 1986, two years after the videogame crash that decimated the industry, Nintendo released the Nintendo Entertainment System (NES) in the United States. The NES helped breathe new life into the videogame industry. Nintendo sold over 62 million consoles before finally discontinuing the NES in 1995.

The NES shipped with a four-direction control pad (see Figure 8.34) featuring two fire buttons as well as Select and Start buttons. These pads can easily be modified to work with the Atari 2600 and other systems that use 2600-compatible controllers, such as Atari 8-bit computers, Commodore Vic-20, and Commodore 64. (See Chapter 7, “Nintendo NES,” for hacks and modifications for the NES console.)

Figure 8.34 NES Control Pad



Preparing for the Hack

The only materials you'll need for this hack are:

- An Atari 2600 joystick (model number CX-40; see Figure 8.35)
- A Nintendo control pad (model number NES-004)

Both are relatively easy to come by—through garage sales, flea markets, thrift stores, shops that sell used game consoles, and, of course, eBay. Thanks to the sheer volume of NES systems sold by Nintendo over the years, finding NES control pads to modify is not difficult at all.

Before performing this hack, you'll want to test both controllers if you can. Specifically, make sure that the four directions and fire buttons work on each controller before taking them apart. This will ensure that the cable from the Atari 2600 joystick is electrically sound and that the direction pad and buttons on the NES controller are without problems.

Figure 8.35 Atari 2600 CX-40 Joystick



The tools required for this hack are:

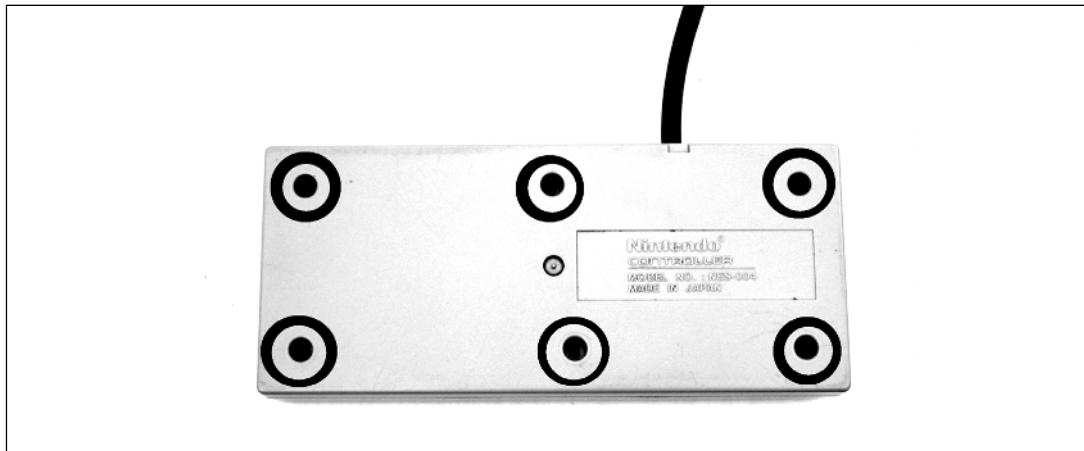
- A Phillips head screwdriver, jeweler's size
- A Phillips head screwdriver, standard size
- Wire cutters and strippers
- Sharp razor blade or Dremel tool
- Soldering iron
- Solder sucker

Performing the Hack

Perform the following:

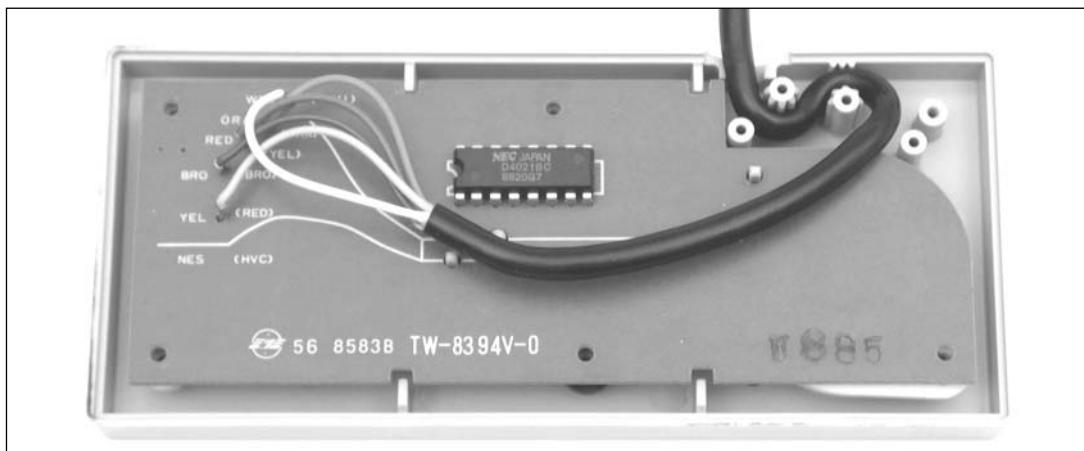
1. Use a jeweler's Phillips head screwdriver to remove the six screws from the bottom of the NES control pad, as shown in Figure 8.36.

Figure 8.36 Bottom of the NES Control Pad



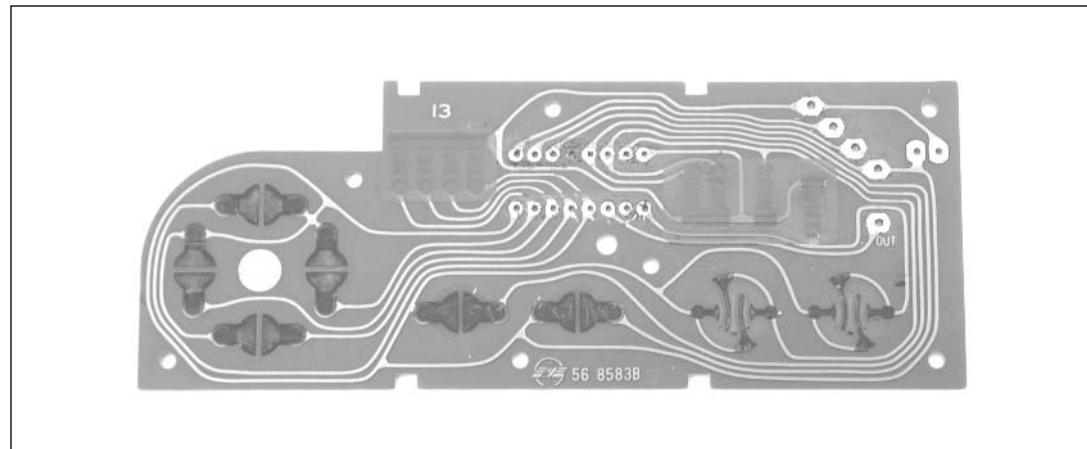
2. Once you have removed the screws, lift up the back half of the control pad. You'll see a simple circuit board, to which five wires and a single IC are attached, as shown in Figure 8.37.

Figure 8.37 Inside the NES Control Pad



3. Remove the circuit board from the case, making sure to leave the rubber button mechanisms resting in the top half of the case. Unsolder the five wires as well as the chip from the board. Be careful not to damage the solder pads for the chip on the opposite side of the board, because this is where we'll be attaching the wires from the Atari 2600 joystick cable. When you are finished, the board should resemble the one shown in Figure 8.38.

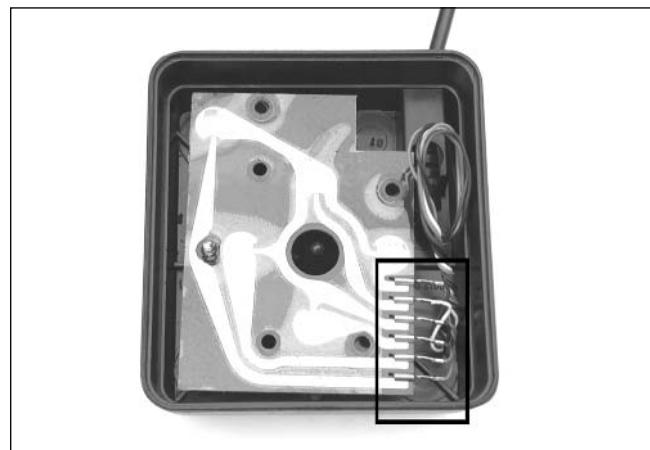
Figure 8.38 Bare NES Control Pad Board



Now it's time to disassemble the Atari 2600 joystick.

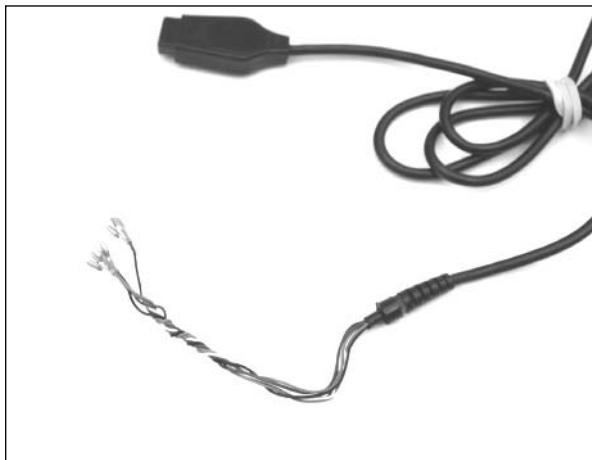
4. Using a standard Phillips head screwdriver, remove the four screws on the bottom of the joystick. Pull apart the two halves of the joystick, revealing a circuit board in the bottom half, to which six wires are attached, as depicted in Figure 8.39. Remove the board from the bottom of the case, and then pull each of the six connectors from the board, as highlighted.

Figure 8.39 Atari 2600 Joystick Circuit Board



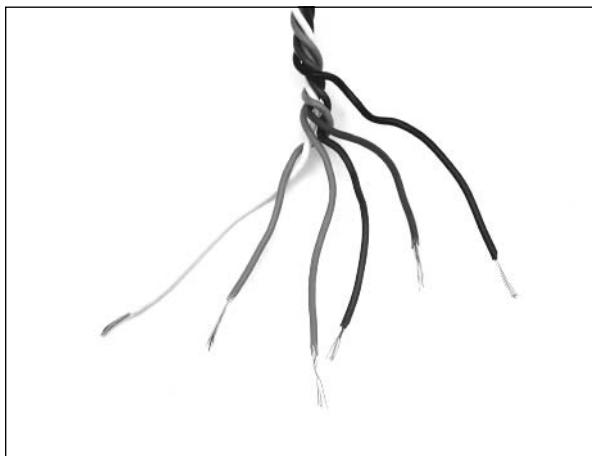
5. With the wires disconnected from the board, you can now remove the cable from the joystick base. We will not need any additional parts from the 2600 joystick, but you can save them for spare parts if you like. Figure 8.40 shows the joystick cable removed from the joystick.

Figure 8.40 Joystick Cable Removed from the Atari Joystick Circuit Board



6. Using a pair of wire cutters, cut the metal connectors from the end of each wire. Then use wire strippers to remove a short length of insulation from each wire (see Figure 8.41).

Figure 8.41 Wires Ready to Be Soldered to the NES Control Pad Board

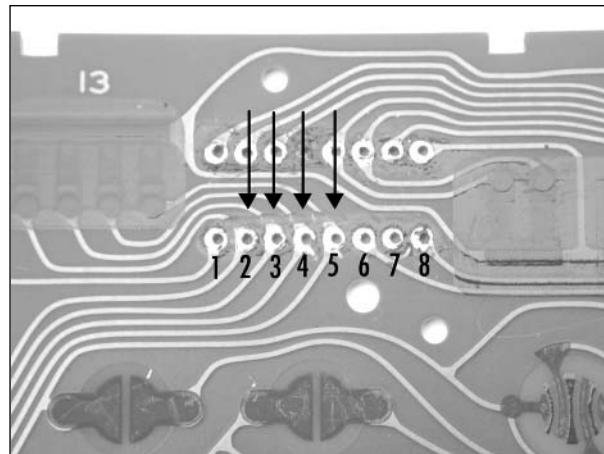


7. Before we solder the wires to the NES control pad circuit board, we must first cut some traces on the board. Using a sharp razor or Dremel tool, cut the traces immediately above the solder pads for holes 2, 3, 4, and 5, as shown in Figure 8.42.

WARNING: HARDWARE HARM

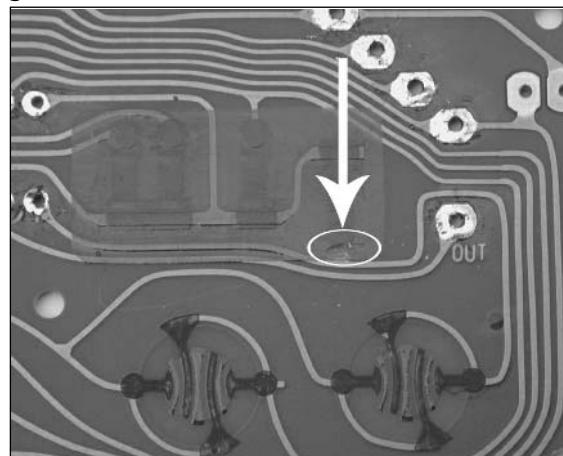
When cutting the traces, make sure that you do not damage the solder pads, because we will be soldering the wires from the Atari 2600 to these pads. Also make sure that you cut the traces above the pads and not below or you will render the circuit board unusable.

Figure 8.42 Cutting the Traces Above the Pads



8. To the right of the holes for the IC are four vertical black strips covered with a green material. Cut the trace immediately below the fourth black strip, as shown in Figure 8.43. Be careful to cut only the vertical trace, not the horizontal trace running below the row of black stripes.

Figure 8.43 Cutting the Vertical Trace Underneath the Black Stripes

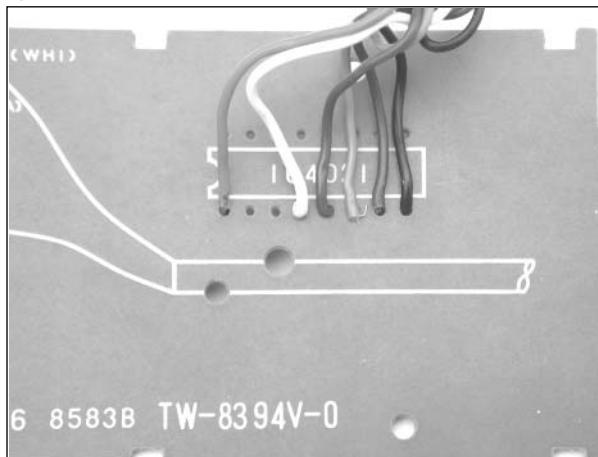


- Now it's time to solder the Atari 2600 joystick cable to the NES control pad circuit board. The individual wires need to be soldered to the bottom row of holes numbered in Figure 8.42, as described in Table 8.3. You might want to "tin" the tip of each wire with solder before inserting them through the holes and soldering them into place.

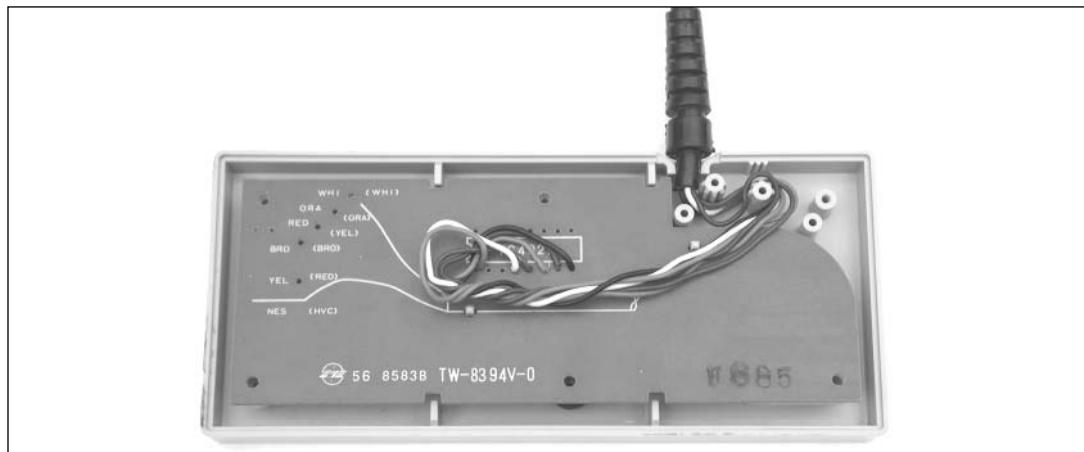
Table 8.3 NES Control Pad Wiring

Hole	Atari Joystick Wire Color
1	Black
2	Brown
3	Green
4	Blue
5	White
8	Orange

- When you're finished, the wires should look as they do in Figure 8.44, as they appear from the opposite side of the board.

Figure 8.44 Wiring Complete

- Now place the solder side of the board down into the bottom half of the controller and thread the cable as shown in Figure 8.45. Be sure that the rubber cups for each button are sitting properly in the bottom half of the controller. If you decided instead to cut the 2600 joystick cable before the stress-relief connector, your wiring will look a bit different (and closer to the original style of the NES control pad cable).

Figure 8.45 Reassembling the Modified NES Control Pad

12. Next, place the back half of the NES control pad cover into place and secure the cover with the six screws removed earlier.

You can test your new controller by plugging it into an Atari 2600 (or another system that accepts Atari 2600-compatible controllers) and enjoying your favorite games (see Figure 8.46).

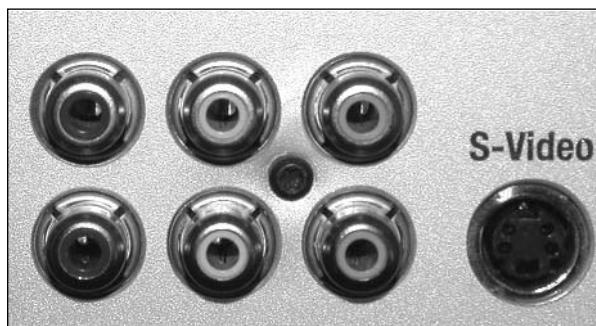
Figure 8.46 NES Control Pad Hack Complete!

Atari 2600 S-Video/Audio Mod

When the Atari 2600 was designed in the late 1970s, the only means of getting a video signal to your television was via the antenna jack. This usually consisted of two screws to which a TV/game switchbox would be connected. A cable then connected the switchbox to the Atari 2600. The signal sent to the television combined the video and audio signals into a single cable, broadcast to the television on a specific channel (in the case of the Atari 2600, channel 2 or 3, selectable via a switch on the console). Unfortunately, this method of transmission introduces significant radio frequency (RF) interference, which is generated by nearby electronic devices (such as the 2600 itself), television stations broadcasting on those and nearby channels, and much more.

Times have changed, and most modern televisions now accept two additional forms of input: composite and S-Video. A composite signal separates the audio and video information onto two separate wires and is free from the typical interference of the older RF method. S-Video is a further improvement in which the video signal is divided into chrominance (more commonly called *chroma*) and luminance (more commonly called *luma*) components, resulting in even further clarity. Modern televisions also have separate inputs for the audio, which are usually in stereo. Figure 8.47 shows audio, composite, and S-Video jacks on a modern television.

Figure 8.47 Audio, Composite, and S-Video Jacks on a Modern Television



Unfortunately, without modification to the Atari 2600, it is impossible to connect it directly to the composite video, S-Video, and audio inputs on your television. However, thanks to a new product by CyberTech, adding such support to the 2600 is now relatively easy. CyberTech has created a board that plugs into the Atari 2600 and allows the console to generate S-Video and stereo audio output. The board results in a much improved picture and higher-fidelity sound.



Preparing for the Hack

Before you can perform this hack, you'll need the following items:

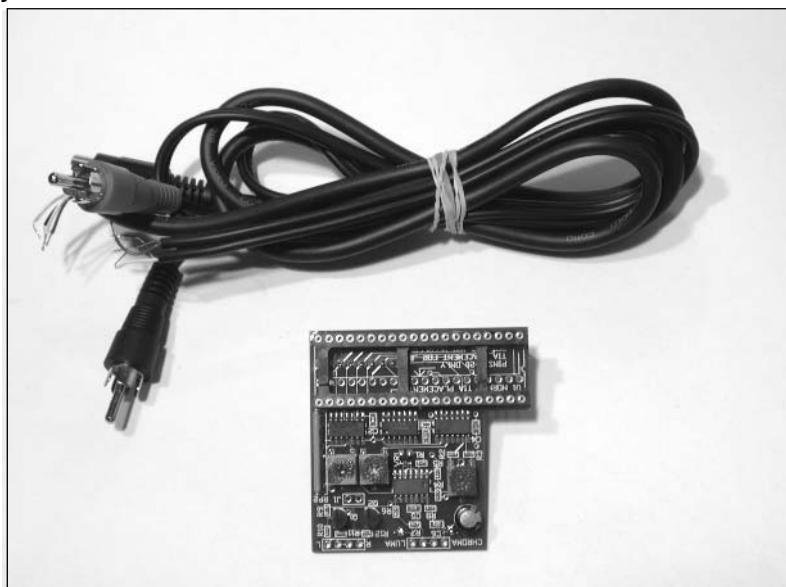
- CyberTech 2600 A/V modification (available from www.Atari2600.com)
- Atari 2600 console (preferably a four-switch unit)

- A 12-foot S-Video cable (optional)
- A 12-foot stereo audio cable (optional)

The CyberTech 2600 A/V modification includes a circuit board, a stereo audio cable, and an S-Video cable (see Figure 8.48). I suggest using a four-switch Atari 2600 unit to perform this modification, since these are the easiest systems to modify and they are also the most common variation of the 2600. This hack is easily reversible because no permanent modifications are made to the console.

As of this writing, the cables included from Atari2600.com are fairly short. You might want to replace them with longer cables to make it easier to hook your 2600 up to your television when you have completed the hack. You can also opt to buy extension cables to extend the reach of your modified 2600, although finding an S-Video extension cable could be tricky. However, Radio Shack sells an S-Video coupler (Part #278-455) that allows you to connect two male S-Video cables together.

Figure 8.48 CyberTech 2600 A/V Modification



The tools required for this hack are:

- A Phillips head screwdriver (to open the 2600 case)
- A small flathead screwdriver (to remove the TIA chip from the Atari 2600 circuit board)
- Needlenose pliers (to remove the metal RF shielding)
- A soldering iron and solder (to connect the A/V cables to the circuit board)
- Wire cutters (to cut the excess leads after soldering the wires to the board)

- A glue gun (optional; you can use a glue gun to secure the audio cables to the 2600 board, to prevent undue stress to the circuit board after the mod is installed)

If you purchase longer audio and S-Video cables to solder to the board, you'll also need:

- Wire strippers (to prepare the wires for attachment to the circuit board)
- A multimeter (to determine how the S-Video cable is wired)

Performing the Hack

Perform the following:

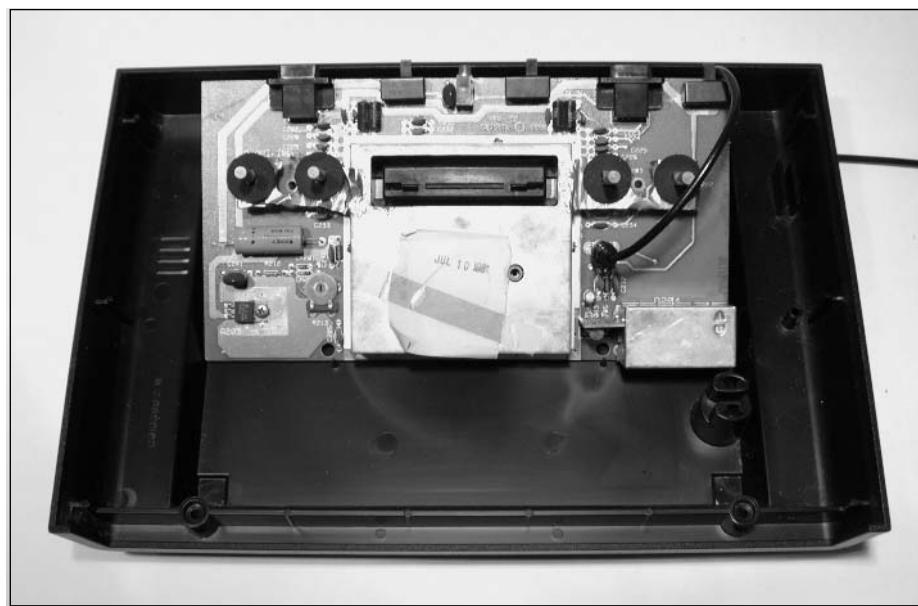
1. The first order of business is to disassemble your Atari 2600. To begin, flip over the 2600 and remove the four screws highlighted in Figure 8.49. When removing the screws, be aware that the two bottom screws are longer than the other two, so you'll want to be sure to insert these into the correct holes when you are reassembling the unit later.

Figure 8.49 The Underside of a Four-Switch Atari 2600



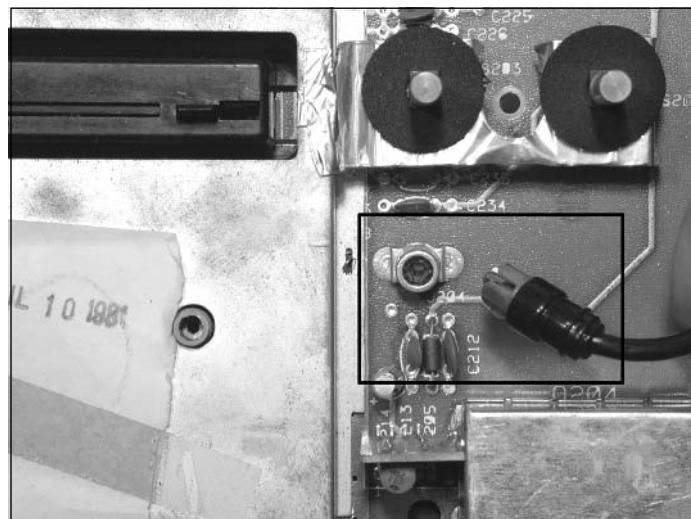
2. Separate the two halves of the case, and put the top half aside for now. If you are using a four-port Atari 2600 as recommended, the internal circuit board should resemble the one shown in Figure 8.50.

Figure 8.50 Atari 2600 Circuit Board with the RF Shield Attached



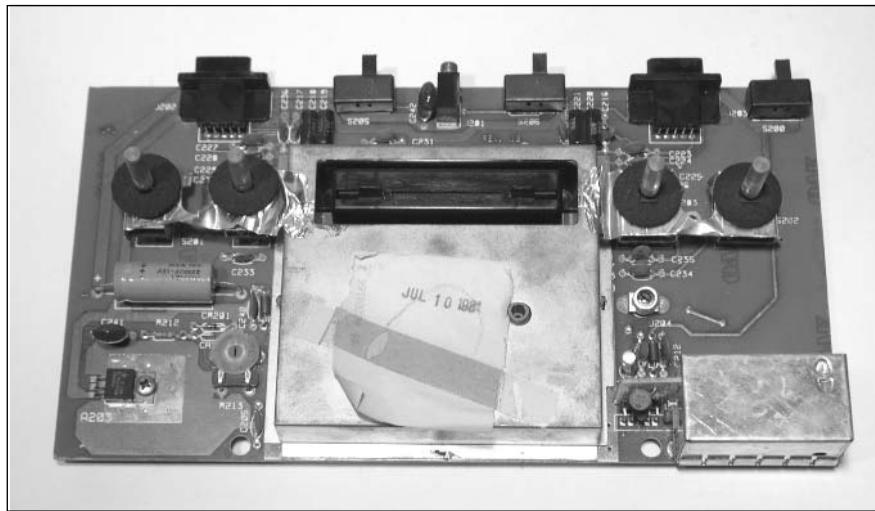
3. Disconnect the RF cable plugged into the circuit board, as shown in Figure 8.51. You can then feed the cable through the hole in the bottom of the case. You can discard this cable, since we won't need it for this hack.

Figure 8.51 Disconnecting the RF Cable



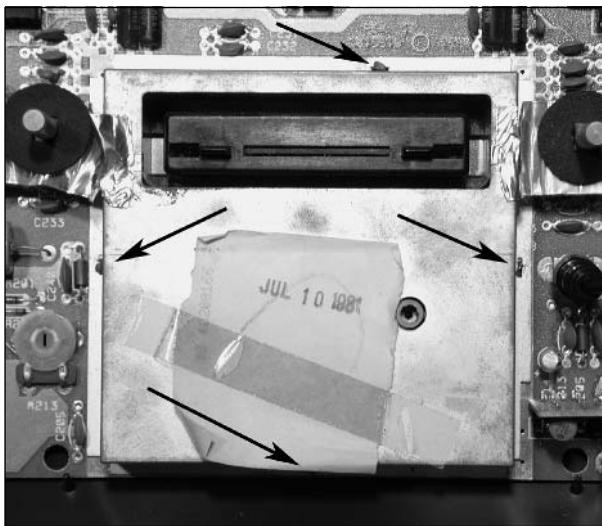
4. Remove the 2600 circuit board from the bottom half of the 2600 case. The board sitting by itself should resemble Figure 8.52.

Figure 8.52 Atari 2600 Circuit Board Removed from the Case



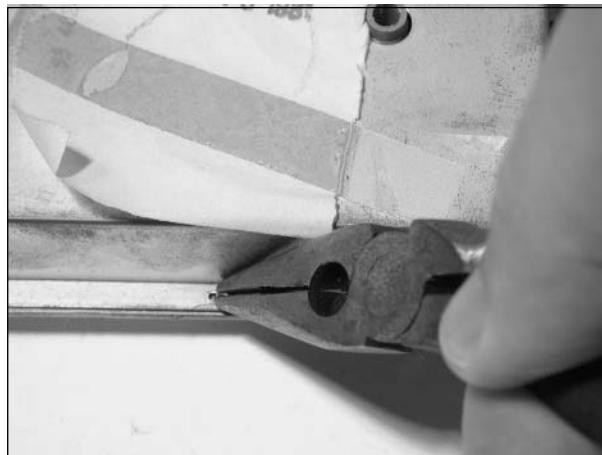
5. Now it's time to remove the RF shielding. The RF shielding consists of two metal halves covering the bulk of the electronic circuitry on the 2600 circuit board. The two halves are held together by metal tabs that must be straightened before the two halves can be pulled apart. Figure 8.53 shows the location of the four metal tabs on this particular version of the 2600.

Figure 8.53 Locations of the Tabs on the RF Shield



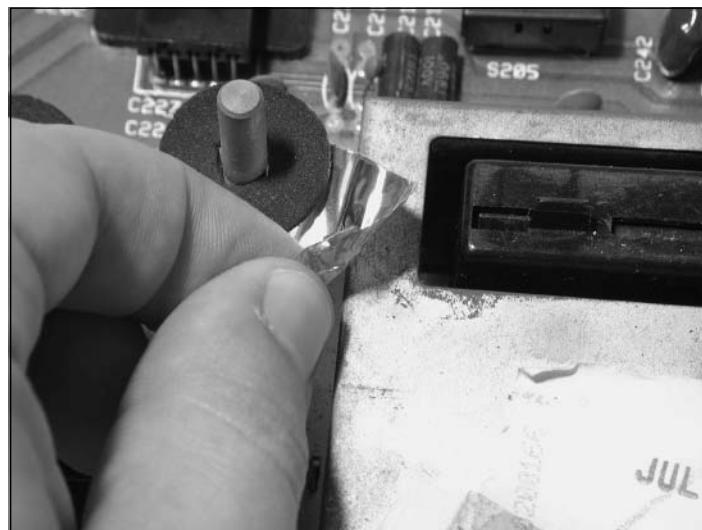
- Using the needlenose pliers, straighten each of these metal tabs so that they are aligned with the slots they are protruding through, as shown in Figure 8.54.

Figure 8.54 Straightening the RF Tabs



- If there is metal tape connecting the 2600 switches to the RF shield, you'll want to peel or cut this tape off, as shown in Figure 8.55.

Figure 8.55 Removing the Metal Tape on the Switches (If Applicable)



- With the tape removed and the tabs of the RF shield straightened, you can now remove the RF shield. The shield will give some resistance, but you should be able to pull it off without

too much effort. You can use a flathead screwdriver to pry between the shield and the board, but be extremely careful that you do not damage any traces or components on the circuit board. After removing the top half of the shield (see Figure 8.56), you can flip the board over and remove the bottom half (see Figure 8.57). We are not going to reattach the RF shield, but you might want to save it in case you decide to reverse this modification at a later date.

Figure 8.56 Removing the Top Half of the RF Shield

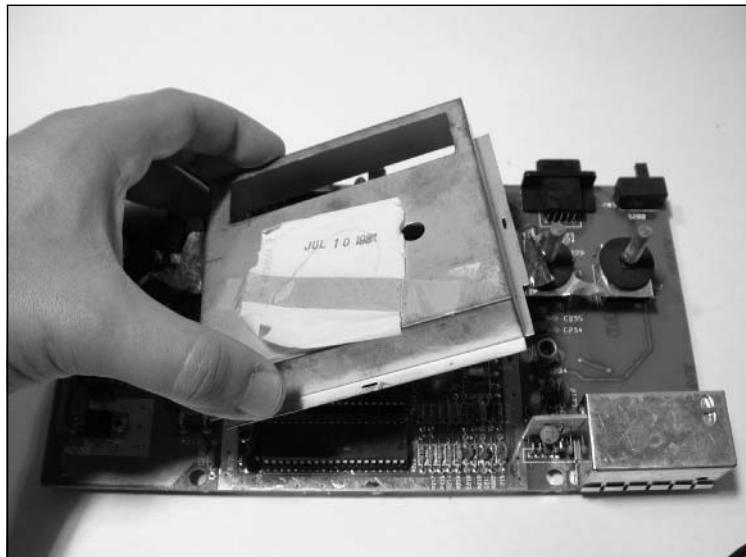
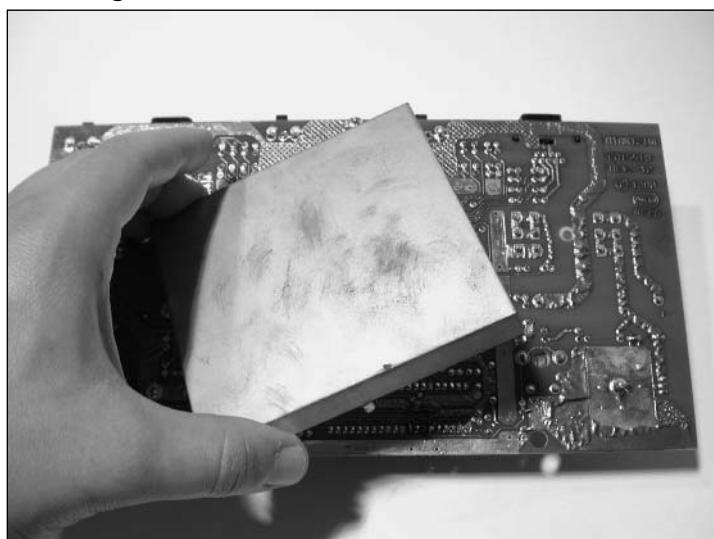


Figure 8.57 Removing the Bottom Half of the RF Shield



9. Next, feed the two cables that we'll be soldering to the CyberTech board through the hole in the bottom of the case (see Figure 8.58).

Figure 8.58 Feeding the A/V Cables into the Case



10. Now it's time to insert the wires into the top side of the CyberTech board. You'll be inserting four wires—audio left, audio right, chroma, and luminance—along with a ground wire for each. Figure 8.59 shows the bottom of the board with the video and audio solder pads highlighted; Table 8.4 lists their descriptions.

Figure 8.59 A/V Solder Pads on the CyberTech Board

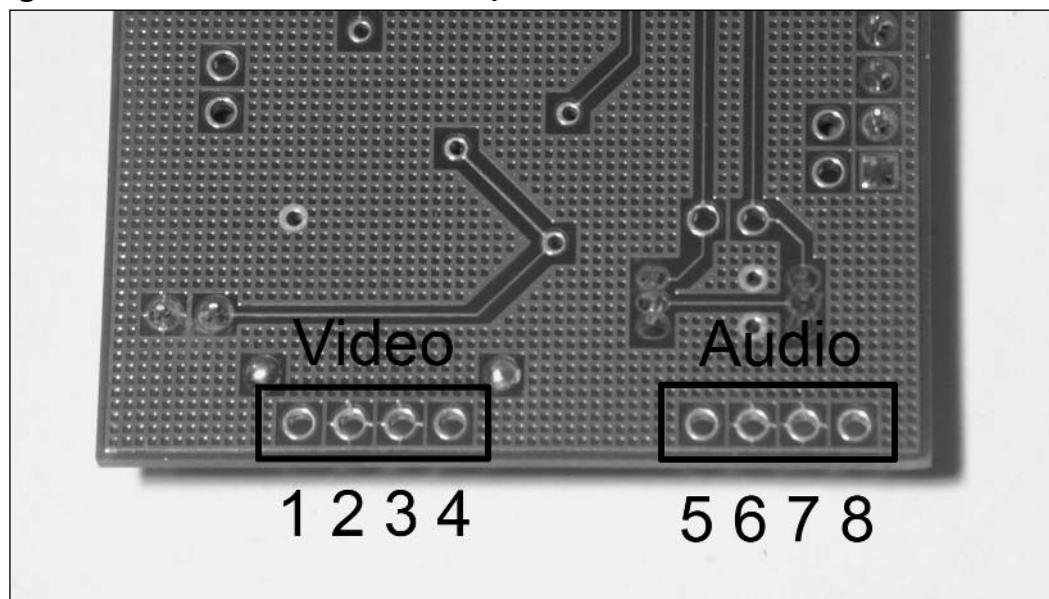


Table 8.4 A/V Solder Pad Pinout and Pin Descriptions

Hole	Description
1	Chroma
2	Chroma Ground
3	Luminance Ground
4	Luminance
5	Audio Right
6	Audio Right Ground
7	Audio Left
8	Audio Left Ground

For the audio cable, insert the red wire into the “Audio Right” hole and the white wire into the “Audio Left” hole. Also, insert the associated ground wires.

The S-Video cable might be trickier. If you are unsure which wire is chroma and which is luminance, you need to use a continuity tester to test the connection between the pins on the S-Video connector and the ends of the wires. Figure 8.60 shows the pinout of a typical S-Video connector, looking at the connector’s male pins. Table 8.5 lists the wire descriptions. For the S-Video cable we used to demonstrate this hack, the red wire was chroma and the yellow wire luma, but this could differ with the cable you are using. It’s best to check in advance to save yourself from having to unsolder and then resolder the wires if you connect them incorrectly.

11. Insert the chroma and luminance wires from your S-Video cable into the CyberTech board.

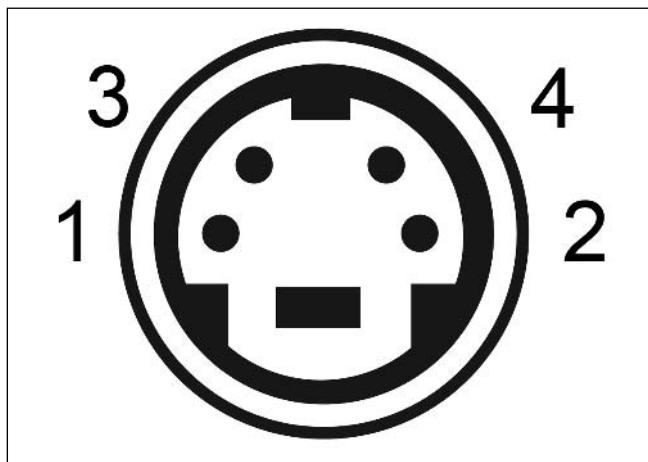
Figure 8.60 S-Video Connector Pinout

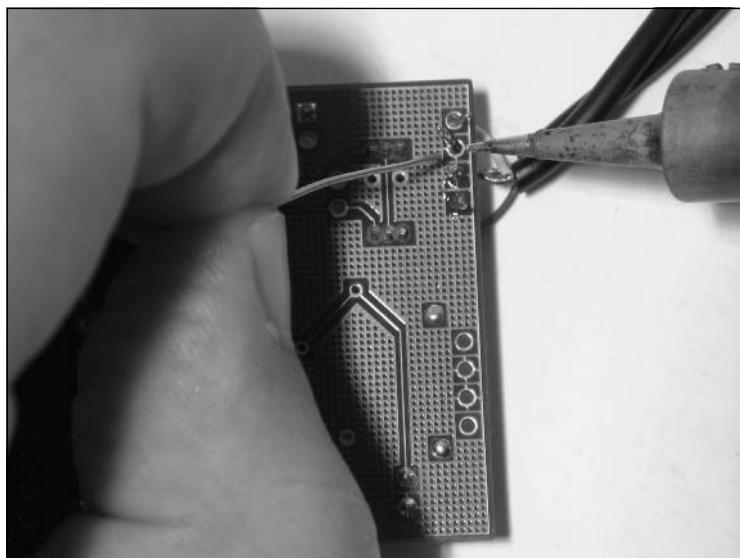
Table 8.5 S-Video Connector Pin Descriptions

Pin	Description
1	Ground (luminance)
2	Ground (chroma)
3	Luminance
4	Chroma

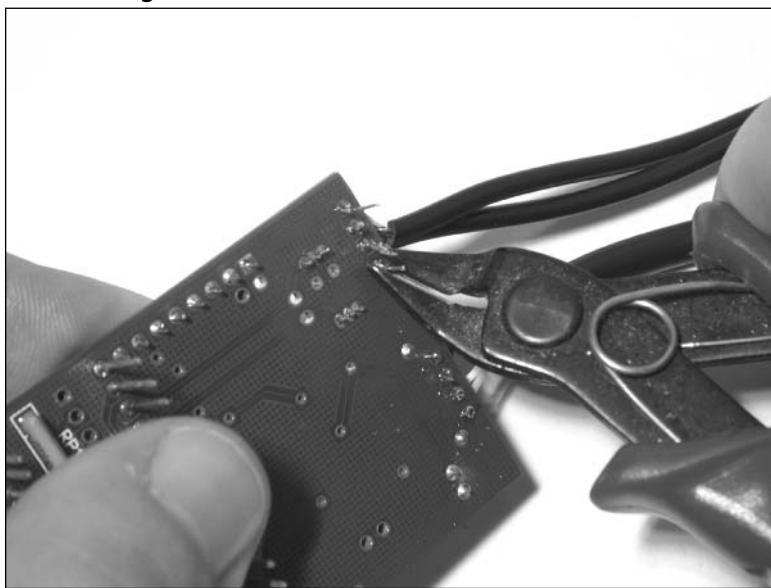
NOTE

Make sure that the S-Video cable has been fed through the hole in the bottom half of the Atari 2600 case before soldering any wires to the CyberTech board. Failure to do so will result in your having to modify the 2600 case later on to fit the connectors through the hole.

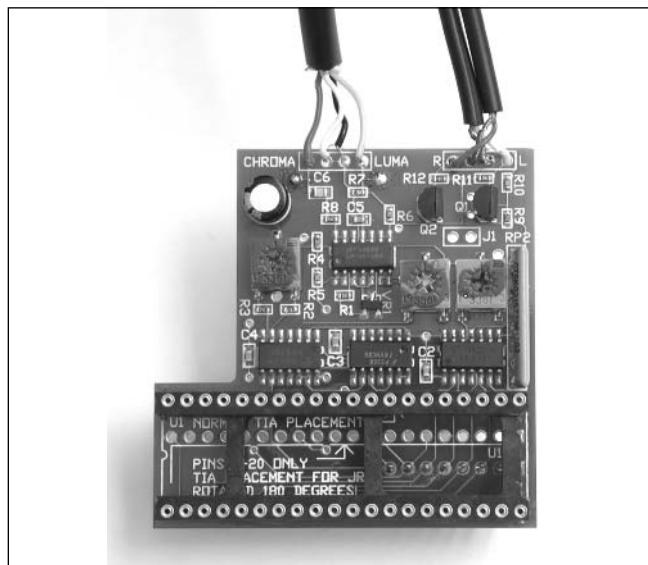
- Now solder the eight connections to the bottom of the board, as shown in Figure 8.61.

Figure 8.61 Soldering the Wires to the CyberTech Circuit Board

- After you have soldered all eight wires, use a pair of wire cutters to cut the excess leads from the board (see Figure 8.62).

Figure 8.62 Removing the Excess Wire Leads

The finished board should resemble Figure 8.63, although the color of your wires may vary, depending on your S-Video cable type.

Figure 8.63 Prepared CyberTech Board Ready for Mounting

14. Now it is time to remove the Television Interface Adapter (TIA) chip from the Atari 2600 circuit board. The CyberTech board uses the audio and video outputs of the TIA chip. We will be removing the TIA chip and placing it into a socket on the CyberTech board. The CyberTech board then plugs into the socket previously occupied by the TIA chip, thus creating a “sandwich.” The location of the TIA chip is highlighted in Figure 8.64.

Using a chip puller or a small, flathead screwdriver, carefully extract the TIA chip from the 2600 circuit board, as shown in Figure 8.65. If you are using a screwdriver, take extra care to not damage the underlying circuit board or any adjacent parts. Furthermore, be careful not to bend the pins on the TIA chip. If the TIA chip is not socketed (and is soldered directly to the board), you need to desolder the chip from the board and replace it with a 40-pin DIP socket.

Figure 8.64 Atari 2600 Circuit Board Showing the Location of the TIA Chip

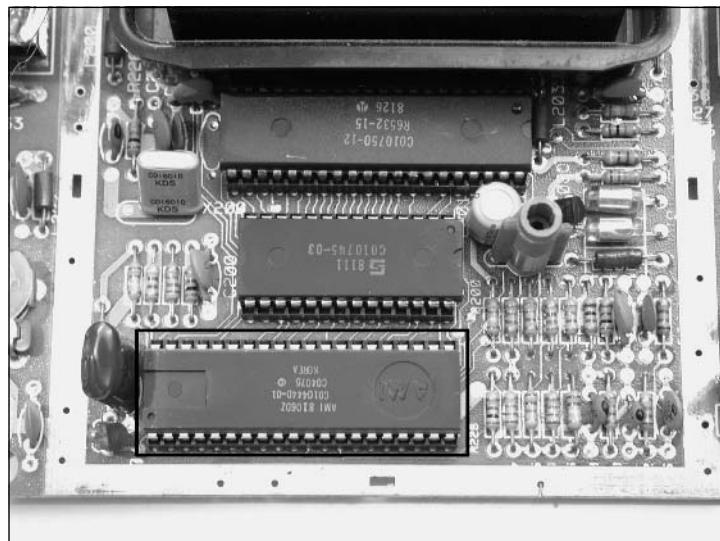
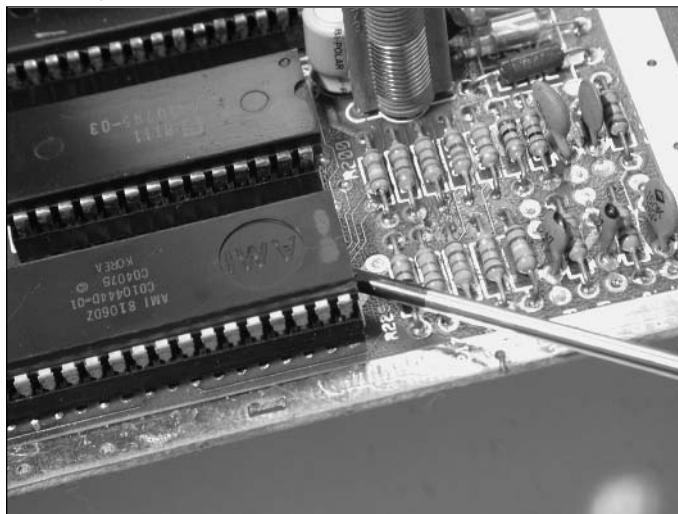
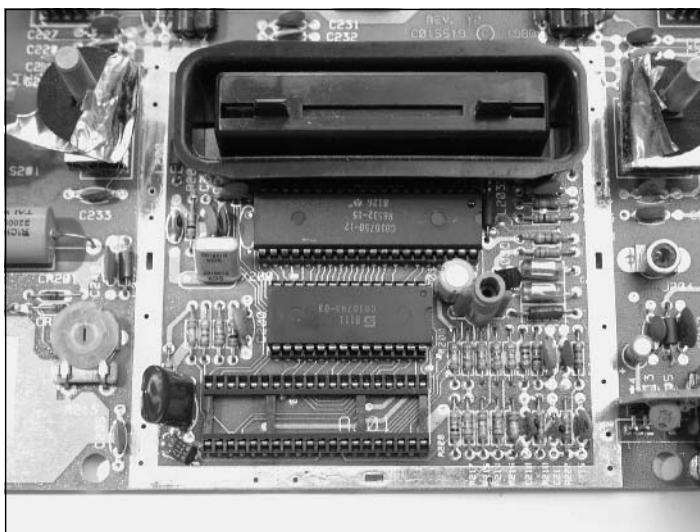
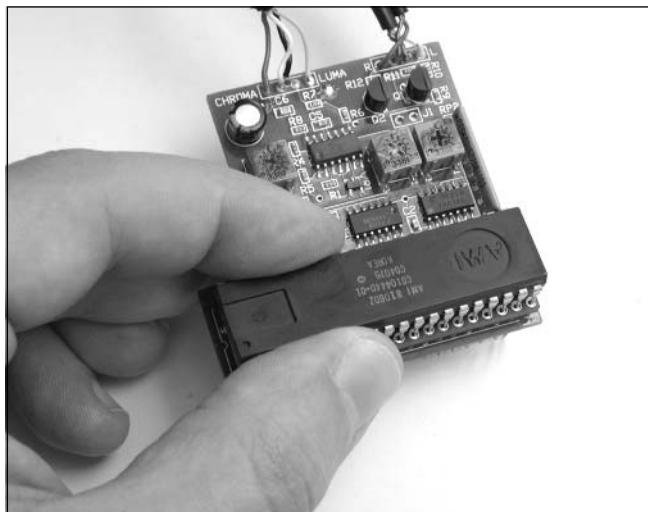


Figure 8.65 Extracting the TIA Chip

15. After removing the TIA chip, your 2600 circuit board should resemble the one shown in Figure 8.66.

Figure 8.66 Atari 2600 Circuit Board with the TIA Chip Removed

16. Next, carefully insert the TIA chip into the socket on the CyberTech board. The notch on the TIA chip must be aligned with the notch visible on the circuit board, as shown in Figure 8.67. Make sure that the TIA chip is firmly seated in the socket.

Figure 8.67 Inserting the TIA Chip into the CyberTech Board

17. Now plug the CyberTech board into the socket previously occupied by the TIA on the 2600 circuit board. First, align all the pins of the CyberTech board with the holes of the socket. Once you are confident that each pin is properly aligned with a hole in the socket, firmly push down on the board to seat it in the socket. You might need to use a fair amount of pressure to properly seat the board, and you should be able to feel when it snaps into place, as shown in Figure 8.68. Ensure that no pins on the CyberTech board are bent or damaged after you've seated the board into the socket.

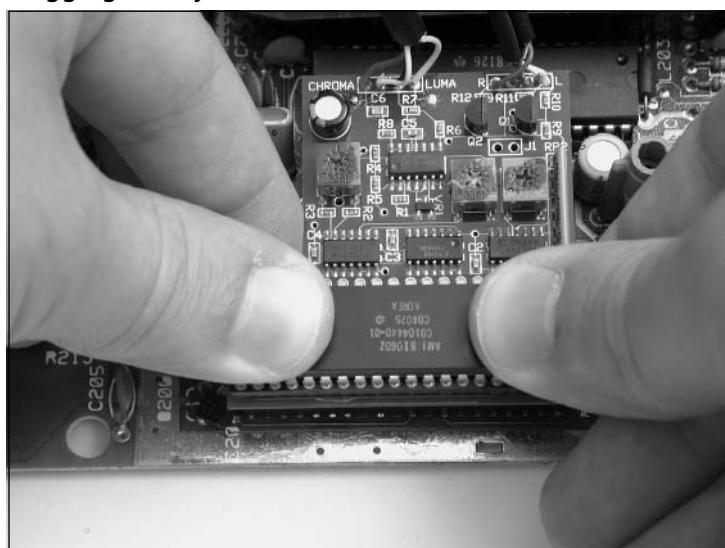
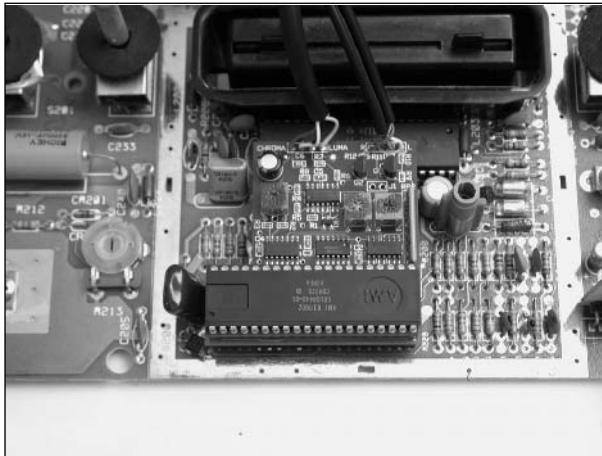
Figure 8.68 Plugging the CyberTech Board into the TIA Socket

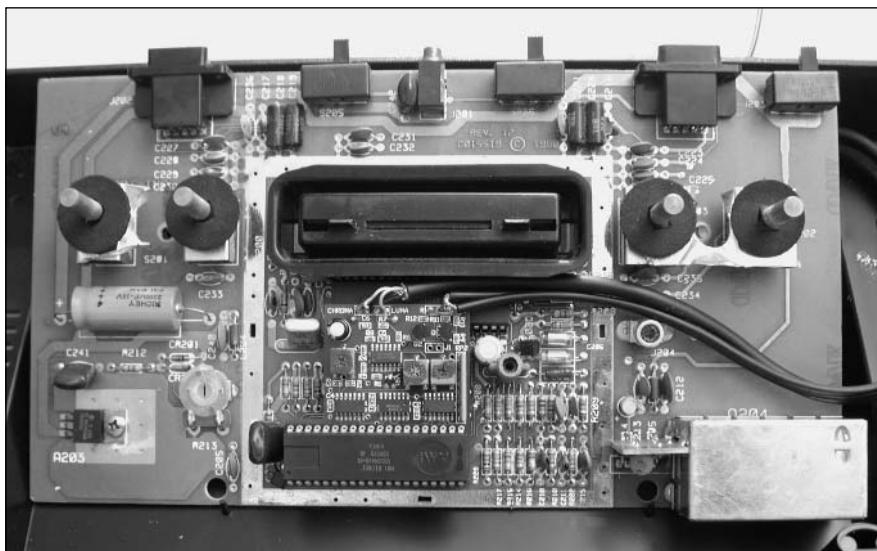
Figure 8.69 shows the CyberTech board resting in its new home.

Figure 8.69 The CyberTech Board Successfully Attached to the Atari 2600

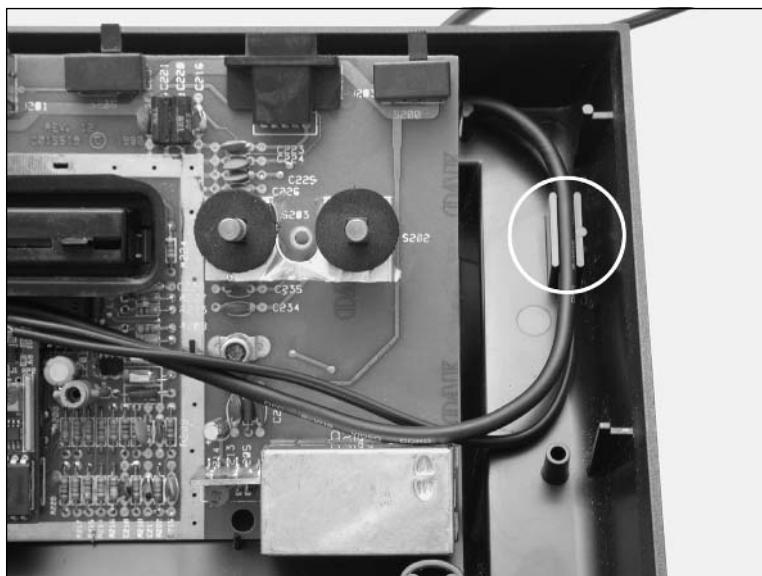


18. Now you can place the 2600 board back into the bottom half of the 2600 case, as shown in Figure 8.70. Feed the cables from the CyberTech board to the right of the case, toward the hole in the back where the cable came in.

Figure 8.70 Securing the Cables



19. Now place the two cables in the cable guide located in the right-back corner of the case, as highlighted in Figure 8.71. This will provide some stress relief for the cables.

Figure 8.71 Securing the Cables, Continued

20. This next step is optional, but if you want to provide some additional stress relief to the cables, you can use a heat gun to secure the cables in place, as shown in Figure 8.72. This will help prevent damage to the 2600 system and CyberTech board in the event the cables are inadvertently yanked.

Figure 8.72 Securing the Cables, Continued (Optional)

21. That's it! Now you can reassemble your Atari 2600. Place the two halves of the case back together, and then fasten the case using the four screws you removed previously. Take care to

insert the correct screw in each hole; with the four-port system we used as our example, the two bottom case screws are longer than the two top screws.

After you've assembled your freshly hacked Atari 2600, go plug it into the S-Video and audio inputs of your television. You should now have a far superior picture than the RF signal you were accustomed to!

Optional: Commodore 1702 Hack

If you have a Commodore 1702 monitor, you can modify the previous procedure slightly to take advantage of the monitor's chroma and luminance inputs. The Commodore 1702 has composite video and audio jacks on the front, but on the back it also has another set of inputs with separate chroma and luminance inputs (see Figure 8.73). This is equivalent to having an S-Video input, but with two RCA connectors instead of the standard S-Video connector that is common on modern televisions.

Due to their excellent picture quality, support for S-Video via the chroma and luminance connectors, convenient carrying handles on the sides, a flat top that allows for items to be placed on top of the monitor, and their reliable nature, Commodore 1702 monitors are popular with classic gaming enthusiasts.

Instead of using an S-Video cable for video output, you'll want to use a high-quality stereo A/V cable that has two RCA connectors on one side (preferably red and yellow, to match the connector colors on the back of the 1702). Solder the red lead of your A/V cable to the chroma solder pad (pin 1 of the CyberTech board). Solder the white lead to the luminance solder pad (pin 4 of the CyberTech board), and solder the two grounds to ground pads on the CyberTech board (pins 2 and 3). This optional hack merely changes the cable type to easily interface to the 1702—the rest of the hack is exactly the same.

When the mod is complete, you can connect the A/V cable to the back of the 1702. Make sure that you also change the Signal Select switch on the back to "REAR." Since the 1702 supports only a mono audio input, you'll need to combine the stereo output from the CyberTech mod into a single signal through means of an RCA Y-Adapter, which can be purchased in many electronics stores.

Figure 8.73 Commodore 1702 Rear Input Jacks



Optional: Do-It-Yourself 2600 A/V Mod

If you don't want to buy an off-the-shelf 2600 A/V mod, such as the CyberTech unit described in this chapter, you can always "roll your own" (which is part of the fun of hardware hacking). You can find plans for several different 2600 A/V modifications on the Internet. Although they will have varying results, they all should generally give you a better picture than the 2600's built-in RF signal.

Nathan Strum has put together an excellent Web page with comparisons of several composite and S-Video mods as well as screenshots showing the differences between them and the 2600's built-in RF output. You can find the page at www.cheeptech.com/2600mods/2600mods.shtml.

If you don't want to modify an Atari 2600 with composite or S-Video output, there is an easy and inexpensive way to improve the RF quality of the 2600. You can replace the TV/game switchbox typically used to connect RF-based systems to a television with a coaxial (F-Type) to female RCA adapter, which you can find at Radio Shack (part #278-276, pictured in Figure 8.74). This allows you to directly connect the RF signal into the cable jack on your television, bypassing the switchbox and generally resulting in a much cleaner picture.

Figure 8.74 Coaxial (F-Type) to Female RCA Adapter



Technical Information

There are four common standards for transmitting a video signal to a television:

- **Component Video** This is a relatively new method employed by the latest televisions, DVD players, and game systems to transmit high-quality video signals. It comprises three signals: one containing black-and-white picture information, or Luminance (Y), and two signals, Cr and Cb, that contain matrixed color information to extract the Red and Blue picture information from the Y signal. The Green picture information is what remains once the Red and Blue information is extracted. This is the preferred choice for hooking up consumer-grade video components. Component video connectors are generally RCA jacks, often labeled Y, Cr, and Cb.

- **S-Video** S-Video output is split into two signals—one containing the black-and-white picture information, Luminance (Y), and another signal, Chrominance (C), that contains all of the color information. S-Video connectors are now standard on most televisions, DVD players, digital video recorders, and other consumer video equipment. An S-Video connector is a small, four-pin mini-DIN. S-Video is sometimes labeled Y-C.
- **Composite Video** Composite video output contains a single signal of all brightness, color, and timing information. Due to this combination of signals, it is impossible for the video to be reconstructed without noticeable artifacts. Composite video generally consists of a single RCA jack, usually yellow to differentiate it from audio jacks (which are generally red and white).
- **RF** In addition to containing the video signal, RF also introduces audio into the signal and is the lowest-quality method of signal distribution to a television. RF is most commonly used to broadcast television signals via over-the-air transmission (picked up with an antenna) or via cable television. RF is capable of carrying a wide range of video and audio signals on different frequencies on a single wire and is prone to interference from many sources. RF jacks on televisions and other equipment usually employ a coaxial F-type adapter. Older televisions might not even have a coaxial jack, instead using two screw terminals to which a switchbox or antenna is attached.

Although component video is beginning to see more frequent use (especially with DVD players), S-Video is still much more common. RF should be avoided whenever possible due to the poor signal quality, although it is difficult to avoid it when working with old videogame systems and vintage computers built in the 1970s and 1980s. Due to the problems inherent to RF signals, hacking old systems to add composite or S-Video output is fairly common and well worth the effort.

Atari 2600 Stereo Audio Output

When Atari introduced the 2600 in 1977, many home videogame consoles (which at the time consisted mostly of Pong machines) had a built-in speaker for audio output. It appears that Atari originally planned to take this same route with the 2600, since the original six-switch model contains two receptacles for mounting small speakers inside the case. Thankfully, Atari decided to send the audio output to the television via the standard RF output instead, so no Atari 2600 models ever shipped with internal speakers. Figures 8.75 and 8.76 show how these speakers were to be mounted in the case.

Speakers? Yes, the Atari 2600 hardware supports left and right audio outputs. Unfortunately, the outputs from the TIA chip are tied together on the circuit board, leaving users with a single, mono audio output. The reason Atari abandoned stereo output is unknown, but it is relatively easy to tap into the individual audio outputs from the TIA and add two RCA ports to your 2600. RCA ports are the standard jacks for making audio connections between audio/video components and are usually white and red, to denote left and right stereo output.

Several Atari games actually benefit from stereo output—most notably, early two-player games such as Combat, Indy 500, Air-Sea Battle, and others. With these games, Player 1's sound effects are

heard in the left sound channel and Player 2's sound effects in the right channel. There's even a new homebrew game for the 2600, Skeleton+, which takes advantage of stereo output to aid the player in locating a skeleton roaming around a 3D maze.

This particular hack will upgrade the audio output for the Atari 2600 from mono to stereo, but you'll still need to get video output by connecting the RF signal to your television. This means that you'll need to use a stereo receiver or similar method to direct the stereo output of the 2600 through speakers, unless your television allows you to change the audio input to a different selection, as some will. This hack will remove the audio output from the RF signal, so you will hear no audio unless you use the newly added stereo jacks.

Figure 8.75 The Top Half of the Original Atari 2600 Case Showing Two Circular "Speaker Grills"

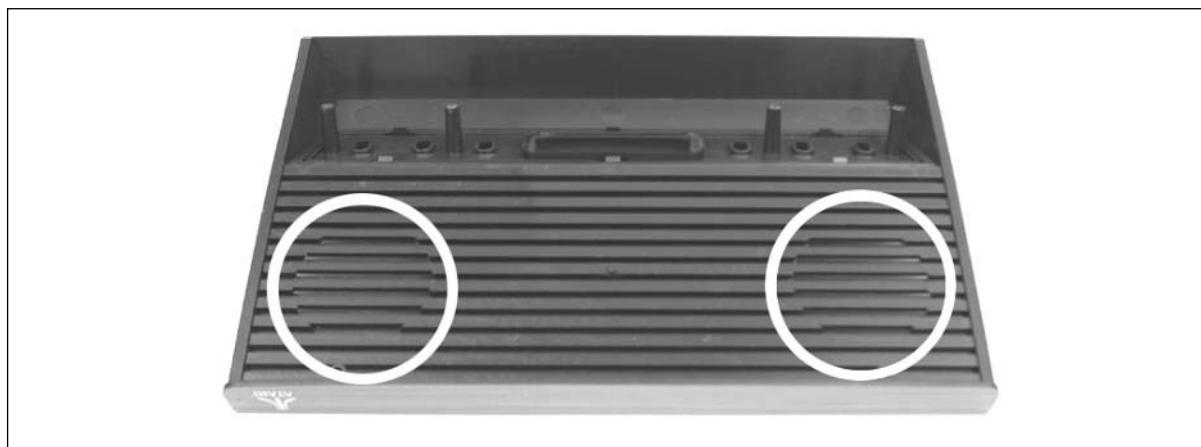
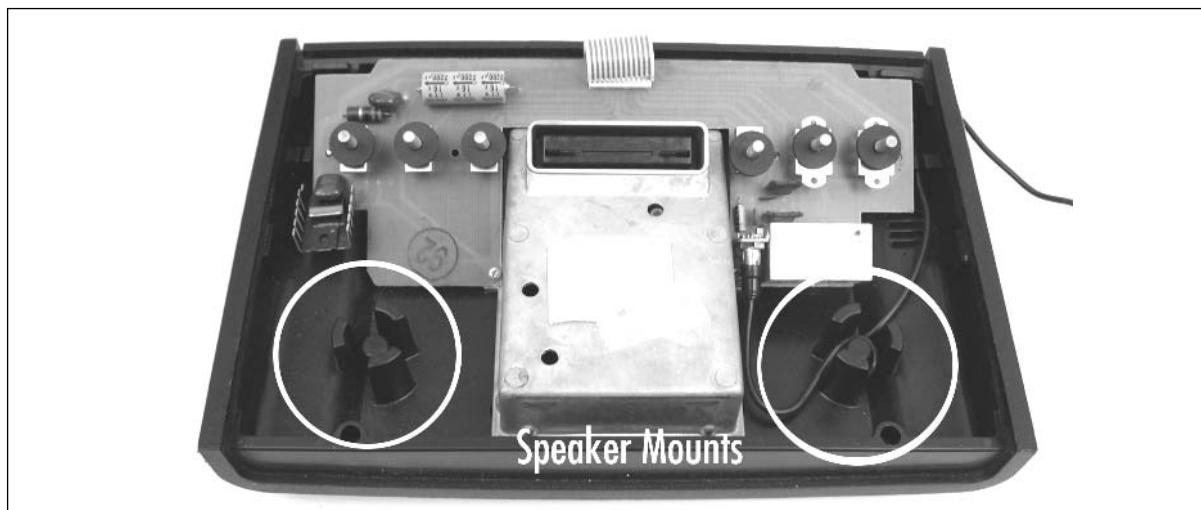


Figure 8.76 Speaker Mounts Inside the Case





Preparing for the Hack

The first thing you'll need is the Atari 2600 system that you want to modify. We will modify a four-switch Atari 2600, since these are the most common. Modifying other versions of the 2600 is a similar process, since they all use the same TIA chip. Most Atari 2600 consoles also use the same basic physical housing, except for the Atari 2600 Jr., which features a much smaller footprint.

The required parts for this hack are:

- A 2.0K ohm, 1% resistor (2)
- A 0.1uF capacitor (2), Radio Shack part #272-109
- An RCA phono jack, female, panel mount (2), Radio Shack part #274-346
- 24 inches of 18 AWG wire
- 2 inches of small diameter heat shrink tubing (optional)

Tools required to perform this hack are:

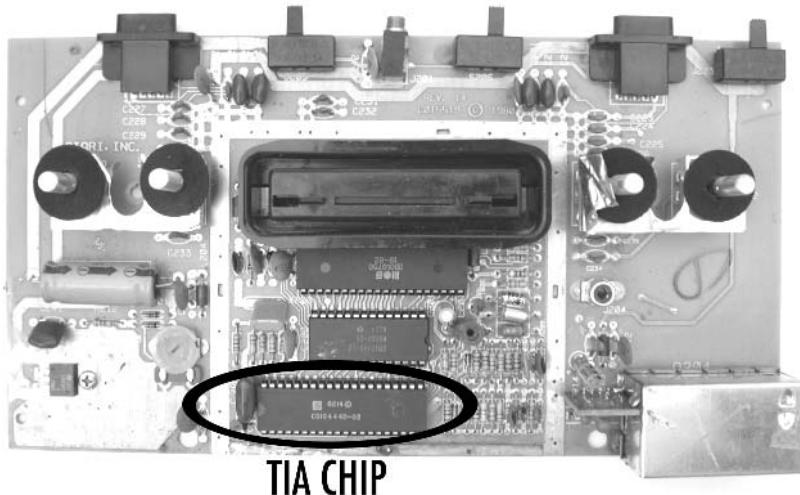
- A Phillips head screwdriver, standard size
- Needlenose pliers
- Wire cutters
- Wire strippers
- A soldering iron
- A drill or Dremel tool with a 1/4-inch drill bit
- A glue gun (optional; you can use a glue gun to secure the audio cables to the 2600 board for a cleaner appearance and to keep the cables out of harm's way)

Performing the Hack

Perform the following:

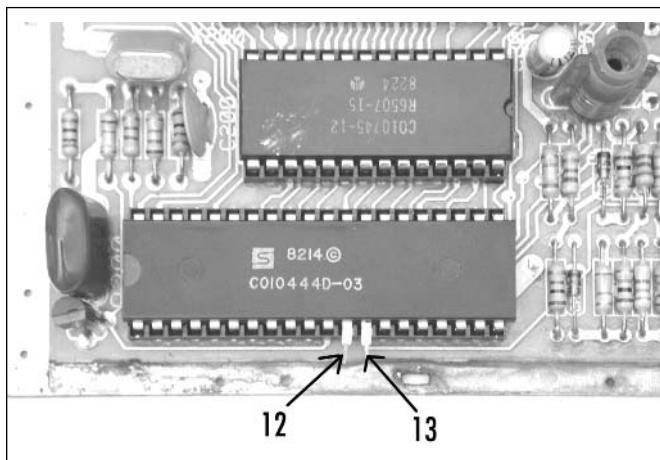
1. The first thing we need to do is open the Atari 2600 and get access to the circuit board. This process is detailed in Steps 1 through 8 of the Atari 2600 S-Video/audio mod presented earlier in this chapter. Be sure to place the RF shield pieces aside for now. The TIA chip is at the bottom of the board, as highlighted in Figure 8.77.

Figure 8.77 Atari 2600 Circuit Board Showing the Location of the TIA Chip

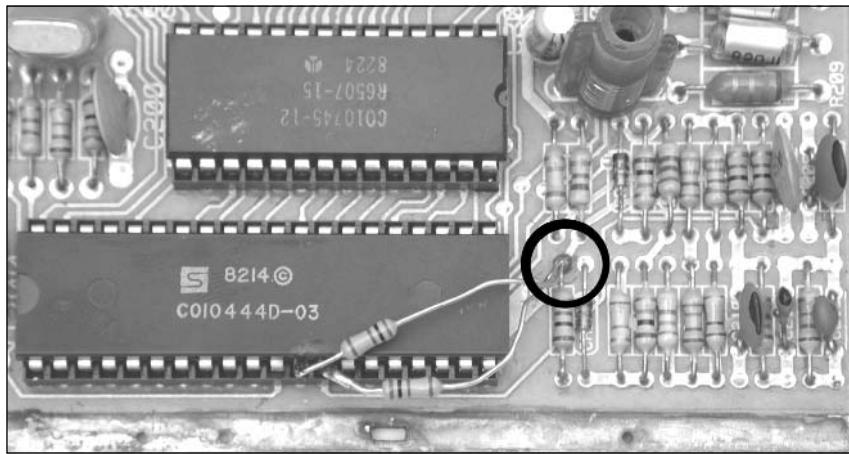


2. If your TIA chip is socketed as shown in Figure 8.77, your job will be easier, since it is a simple matter to remove the chip and bend the two pins we'll need to access. Otherwise, you need to carefully cut the two pins, which is somewhat trickier. What we're after are pins 12 and 13, which represent the two audio channels of the 2600.

If the TIA chip is socketed, remove it and bend pins 12 and 13 upward, as shown in Figure 8.78. Then reinsert the chip into the socket, taking care to note the proper orientation of the chip. If the chip is soldered to the board, you can use a Dremel tool to cut the pins as close to the board as possible and then bend them upward.

Figure 8.78 TIA Chip with Pins 12 and 13 Bent Upward

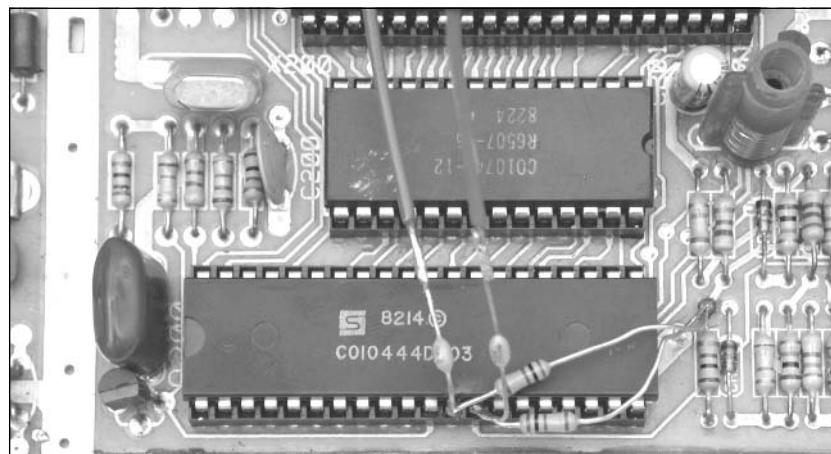
- Now warm up your soldering iron and solder the two 2.0K ohm resistors from pins 12 and 13 to the 5V solder pad shown in Figure 8.79.

Figure 8.79 Resistors Soldered to the TIA

- The next step is to prepare the two wires that you will be soldering to the audio jacks on the back of the 2600 case. Cut two lengths of the 18 AWG wire, each about a foot long. These wires will run from the TIA chip to the back of the case. Solder one side of the 0.1uF capacitor to each of these cables. You might want to cut a 1-inch or so length of the 1/32-inch heat-shrink tubing and slide it over each of these cables—these will help shield the connections you make to the TIA chip. If you don't have heat-shrink tubing, you can use electrical tape to insulate the connections, but it's important that the connections are shielded in some fashion to prevent shorts from occurring when the RF shield is replaced.

Now take the opposite end of each capacitor and solder one each to the joints formed by pins 12 and 13 and the two resistors you previously soldered, as shown in Figure 8.80.

Figure 8.80 Capacitors Soldered to the TIA

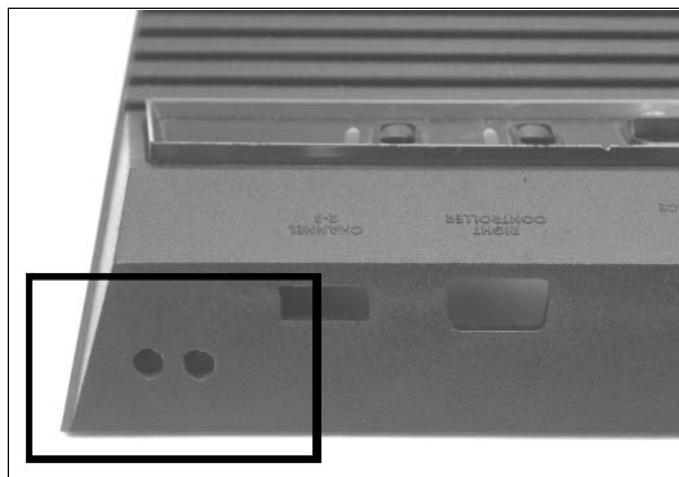


5. We now have to prepare the 2600 case to mount the two RCA jacks. The easiest place to mount the jacks is along the back side of the top half of the case. This is the same side that the RF cable resides on. Using a drill or Dremel tool with a 1/4-inch drill bit, drill the two holes (see Figure 8.81).

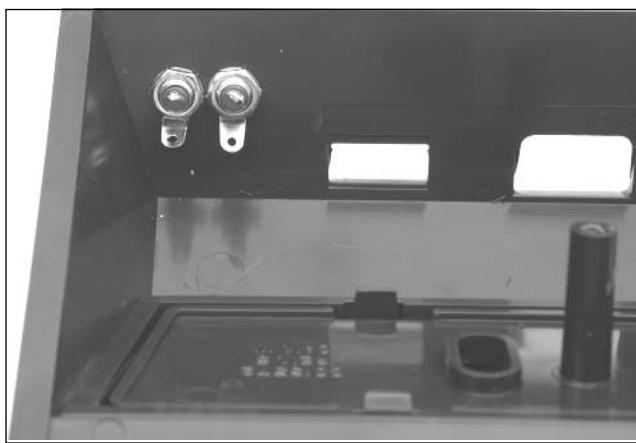
WARNING: HARDWARE HARM



Do not place the holes too close to the side of the case or too low, or they will be difficult to access from either inside the case or outside (when the top half of the 2600 case sits on the bottom half, there is a lip that overlaps). In addition, space the holes sufficiently apart so that the two jacks are not too close to one another.

Figure 8.81 Holes Drilled in the 2600 Case for the RCA Jacks

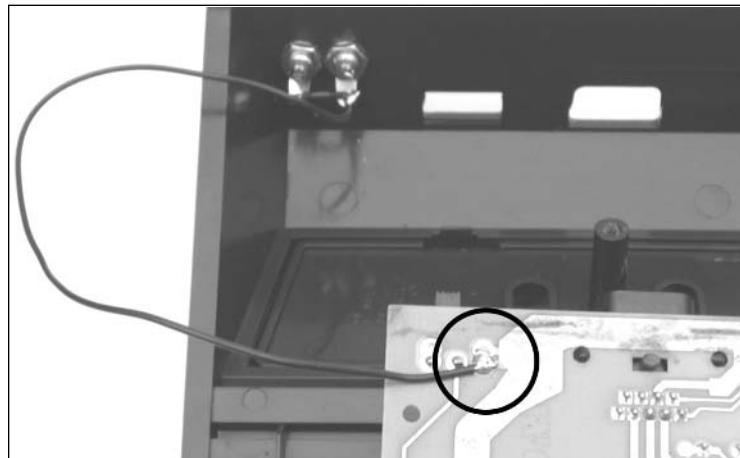
6. Once the holes are drilled, you can attach the two RCA jacks. They should comprise three pieces: the jack itself, a small washer, and a nut. For each jack, remove the washer and nut, and push the threaded side of the jack through from the outside of the case. Then slide the washer over the jack and screw the nut into place. When attached, the jacks should resemble Figure 8.82.

Figure 8.82 RCA Jacks Attached to the 2600 Case

7. With the jacks attached, you can now solder the last few connections. The tab on the outside ring of each jack needs to be soldered to ground. Join these two tabs with a piece of wire (preferably black so that it is easy to denote as a ground connection), and then, using a separate length of wire, solder one of the tabs to an available ground on the circuit board. If

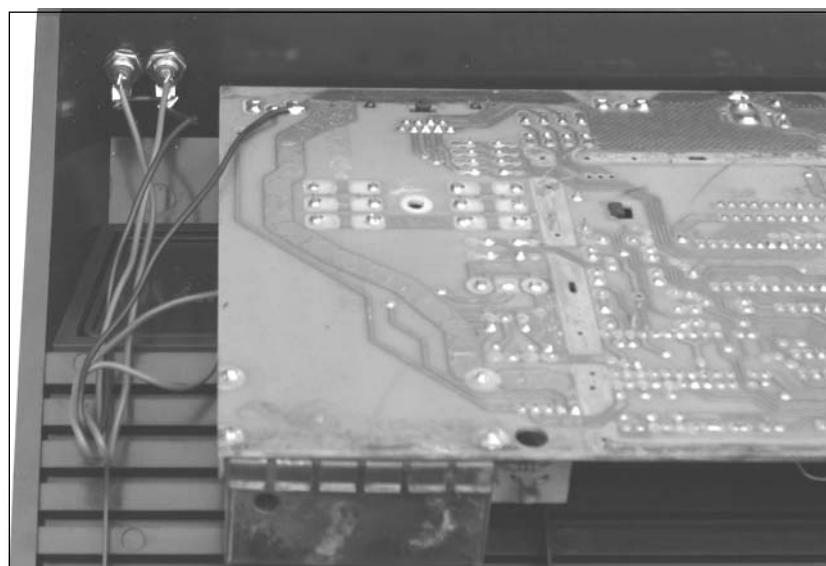
you flip the board over, you'll see that the most convenient solder point is connected to the channel 3/4 switch. Solder the wire to this location, as shown in Figure 8.83.

Figure 8.83 Solder the RCA Phono Jacks to Ground



8. The only two remaining connections are for the wires you soldered to the TIA chip. Solder one wire each to the center post of each RCA jack, as shown in Figure 8.84.

Figure 8.84 Solder the Wires from the TIA to the RCA Jacks



9. You're almost done! Your completed hack should look similar to Figure 8.85. You can use a glue gun to securely attach the wires to the circuit board to make sure that they don't get in the way of the cartridge port or switches.

Figure 8.85 Completed Stereo Audio Output Hack



10. Reassemble the 2600 by replacing the RF shield, plugging the RF cable into the board, placing the board back in the case, and screwing the case back together. (Remember, the long screws are for the bottom holes.)

Once your system is assembled (see Figure 8.86), hook your 2600 up to your television and stereo system and fire up a game of Combat. You should distinctly hear Player 1's tank in one speaker and Player 2's tank in the other. Try some other games as well—you might be surprised at the stereo effects you find!

If you'd like to try your hand at Skeleton+, a modern homebrew game programmed to take advantage of stereo output, you can purchase the game in cartridge form at AtariAge (www.atariage.com/store). You can also download the binary from AtariAge so that you can try it in your favorite Atari 2600 emulator before buying—the author has made it freely available for anyone to download.

Figure 8.86 Stereo Jacks on the Outside of the Case



Under the Hood: How the Hack Works

The specialized TIA chip in the Atari 2600 supports two audio channels, which are output separately on different pins exiting the chip. As soon as these pins connect to the circuit board, they are tied together, resulting in a monaural output from your television. Because the signals for these two channels exit the TIA separately, we can tap into them and direct the signals elsewhere, as we have done with this hack.

Many games are stereo in nature, due only to the fact that the Atari 2600 has two audio channels. Many of Atari's earlier games used one channel for Player 1's sound effects and the second channel for Player 2's sound effects. This results in a very noticeable and useful stereo effect. A recent homebrew game for the 2600, Skeleton+, intentionally uses the stereo effect to help direct the player to his foe in a 3D maze. However, with some games the stereo effect may be less desirable, as is the case with Pitfall II, where the (excellent) music plays in one channel and the sound effects in the other.

A useful addition to this hack is to add a switch that allows you to toggle between stereo output and mono output, allowing you to enjoy games like Pitfall II without having to deal with potentially annoying stereo sound effects.

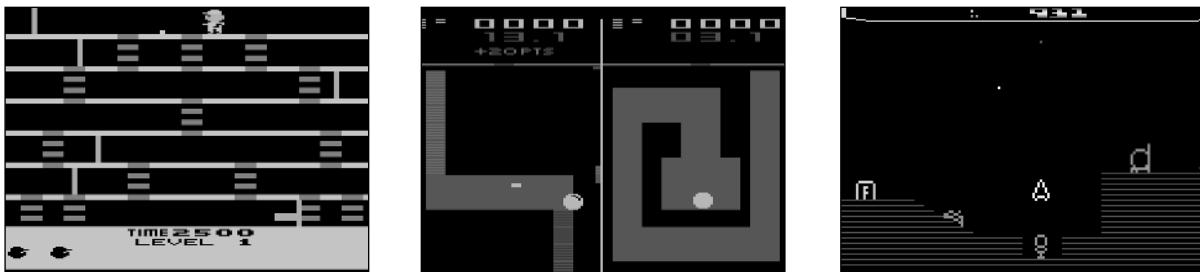
Homebrew Game Development

The Atari 2600, introduced in 1977, is nearly 30 years old, but it has a following of loyal fans that are creating new games for it today. Even though hobbyists write these games in their spare time, they typically equal or surpass the quality of the commercial games released for the Atari 2600 back when the system was still in its prime. Many of these games, typically known as *homebrew games*, have been released for the Atari 2600 in the last several years. Figure 8.87 shows a small selection of the homebrew games available for purchase at the AtariAge booth during the Midwest Gaming Classic in May 2004.

Figure 8.87 Homebrew Games For Sale at a Recent Video Game Show

Homebrew authors today have many advantages over programmers who were writing games for the Atari 2600 back in the late 1970s and early 1980s. Thanks to the Internet, a vast warehouse of information on how to program the Atari 2600 is available online and easily within grasp of anyone who can wield a search engine. This same information was once a closely guarded secret, when companies such as Atari and Activision competed fiercely in the marketplace and were always looking for a technical advantage to give their games an edge over their competitors.

In addition to many Web sites with technical information on how to program the Atari 2600, you can also converse online with experienced 2600 programmers. These programmers are quick to share their knowledge and offer insight on how to improve your code and make your game more enjoyable. Some homebrew authors have even embarked on writing detailed tutorials for those who want to get their feet wet in Atari 2600 programming. In addition to talking with fellow authors online, many programmers also solicit feedback from Atari 2600 fans while they are developing their game. Figure 8.88 shows screenshots from three excellent 2600 homebrew games: Climber 5, Marble Craze, and Thrust+ Platinum.

Figure 8.88 Screenshots from the Homebrew Games Climber 5, Marble Craze, and Thrust+ Platinum

Another advantage of homebrew authors today is excellent development tools. First-class Atari 2600 emulation allows authors to instantly test their creations on their personal computer. Flash ROM-based cartridges allow programs to be downloaded to a cartridge that can be used to test software on a real Atari 2600, without having to do any soldering. Development software is also plentiful, allowing quick compilation and generation of Atari 2600-compatible binaries.

Two hardware tools that have been useful to Atari 2600 programmers are the Supercharger and Cuttle Cart. The Supercharger is a peripheral sold by Starpath (originally Arcadia) in the early 1980s; it allows you to load games from cassette tapes. Since the Supercharger needs only an audio source to load games and doesn't care what the actual media is, you can load games from an audio CD or even directly from your personal computer. Although only intended to load commercial games sold by Starpath, the Supercharger can be modified to load any 2K or 4K game. Information on how to perform this modification can be found at www.atari2600collector.com/scmod.htm. Used Superchargers can be found relatively inexpensively, making the Supercharger an excellent investment for people who want to develop 2600 games.

The Cuttle Cart (see Figure 8.89) is a modern Atari 2600 cartridge created by Chad Schell of Schell's Electronics (www.schells.com) that uses the same principle as the Supercharger to load games. However, unlike the Supercharger, the Cuttle Cart supports nearly every bankswitching method employed by 2600 games. This allows the vast majority of games to be played using the Cuttle Cart, making it a more flexible development tool, since only games up to 4K in size can be run on the Supercharger. Unfortunately, the Cuttle Cart is no longer being produced (about 200 were manufactured), but they are periodically offered for sale by their owners. Chad Schell has created a new version of the Cuttle Cart for the Atari 7800, which supports both 2600 and 7800 games, loads games from a Multimedia Card (MMC), and has sophisticated onboard menu software.

Figure 8.89 Cuttle Cart by Schell's Electronics



After you have completed a game, it is easy to get it produced in cartridge form. You can choose to buy all the components, build cartridges and sell them yourself, or let a third party publish your titles and draw royalties from sales of the game. You won't become rich selling Atari 2600 titles, but it is very satisfying to see your title in an online store—complete with a professionally designed label and manual.

Building a new Atari 2600 cartridge from scratch is much easier today than in the past, thanks to the availability of new Atari 2600 printed circuit boards (PCBs), such as those produced by Pixels Past and sold by AtariAge. Before new circuit boards were available, it was necessary to hack apart and modify existing Atari 2600 game PCBs—a tedious process, since the existing ROM chip needed to be unsoldered from the board. With new PCBs, all you need to do is purchase the various parts, program the EPROM with your game binary, and then solder the parts to the board.

After the board has been tested, it can then be placed into an Atari 2600 cartridge shell, a new label can be applied, and you have a new game that can be enjoyed on a real system! Figure 8.90 shows a Pixels Past bankswitch 8K/16K/32K Atari 2600 PCB bare (on the left) and both sides of a completed board (center and right). Parts are soldered onto both sides of the PCB, which is necessary to keep the circuit board size the same as Atari's, allowing these new PCBs to be used in existing Atari cartridge shells.

Figure 8.90 New Atari 2600 PCBs by Pixels Past

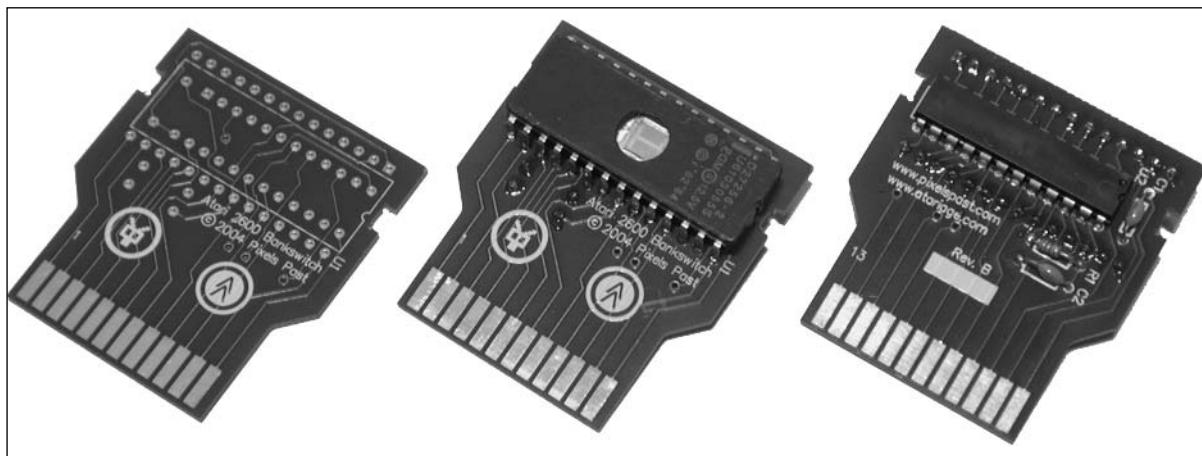


Figure 8.91 shows a completely assembled Atari 2600 homebrew game as sold by AtariAge. This particular example, Star Fire, also includes an original, high-quality, and professionally printed label and manual, putting these new releases on par with games released by Atari over 20 years ago.

Figure 8.91 Atari 2600 Homebrew Star Fire



Writing Atari 2600 games is very challenging, but thanks to the Internet, it's not the nearly impossible task it used to be for homebrew authors. If you'd like to learn more about creating homebrew games for the Atari 2600, the following links will serve as good starting points:

- **AtariAge, www.atariage.com** AtariAge is a staunch supporter of the Atari 2600 homebrew scene and offers many resources for the homebrew author. The first is its In Development section, which lists all the 2600 games currently in development, along with descriptions and screenshots. Next are the Homebrew Development forums, which are frequented by many homebrew authors and a great way for authors to get help and feedback while developing a game. When your game is complete, AtariAge can publish your title and offer it for sale in its online store at www.atariage.com/store. AtariAge is the largest publisher of Atari 2600 homebrew games and frequents many classic gaming events to further promote the homebrew scene. AtariAge also sells Atari 2600 circuit boards created by Pixels Past (www.pixelpast.com) and other components for authors who want to build their own cartridges.
- **Stella Mailing List, www.biglist.com/lists/stella** The Stella Atari 2600 Programming List is a mailing list dedicated to programming the Atari 2600. Frequented by a large number of active Atari 2600 programmers, it is one of the best sources of information when you need an answer to a tough question. Also be sure to check out the extensive and searchable archives.
- **2600 Programming for Newbies, www.atariage.com/forums/viewforum.php?f=31** Atari 2600 programmer Andrew Davie (author of *Qb*) is writing a constantly evolving programming tutorial for those who know little about programming the Atari 2600 but would

like to dive into it. Andrew's tutorial is broken into a series of lessons, and he gladly answers any questions asked. As of this writing, 24 extensive sessions are available, providing a great introduction to 2600 programming.

- **2600 101, <http://alienbill.com/2600/101>** An excellent annotated eight-step tutorial for everyone who wants to give programming the 2600 a serious try. This site should guide you safely through all the early traps. This tutorial was written by Kirk Israel, who released the Atari 2600 homebrew game JoustPong in the spring of 2004.
- **DASM Assembler, www.atari2600.org/dasm** DASM is a versatile macro assembler, with support for target microprocessors, including the 6502 and 6507 (the core used in the Atari 2600 console). It is the standard assembler for all 2600 development and it is strongly suggested for 2600 programmers to download and use the VCS.H and MACRO.H files as well to match the common Stella Mailing List standards.
- **z26 Atari 2600 Emulator, www.whimsey.com/z26/z26.html** z26 is an excellent Atari 2600 emulator, preferred by many homebrew authors for its accuracy in simulating a real 2600. Windows and DOS versions are available, and the program is updated frequently.
- **Atari 2600 Programming Resources, www.atariage.com/2600/programming** AtariAge has put together a page of useful links to Atari 2600-related documentation, tutorials, tools, utilities, source code, hacking, and more.

Atari 2600 Resources on the Web

You'll find a great number of resources on the Web for enthusiasts of the Atari 2600. Whether you're looking for information about your favorite games, how to hack your system to give it capabilities never intended by the designers, or forums where you can discuss the Atari 2600 with other fans, you're sure to find it online. Some quick hunting with your favorite search engine will reveal many Atari 2600 sites, but we'll get you started with a brief list of our favorites:

- **AtariAge, www.atariage.com** Evolved from The Atari 2600 Nexus in 2001, AtariAge contains a wide variety of information about Atari game systems, featuring thousands of scans of 2600, 5200, 7800, Jaguar, and Lynx games, manuals, and boxes. In addition, you'll find the latest Atari-related news, updates on new homebrew games in development, an online store featuring classic gaming hardware and software, and an active community discussing Atari systems in online forums.
- **Atari History Museum, www.atarimuseum.com** The Atari History Museum is a comprehensive Web site covering the entire range of Atari's history, from its early days of Pong to the days when the Atari Jaguar was state of the art. You'll find a huge wealth of historical information and pictures about Atari game consoles, computers, arcade games, and much more. The Atari History Museum got its start on the Web in 1997 and has been

adding and expanding ever since. Its curator, Curt Vendel, originally ran a bulletin board system in the 1980s dedicated to Atari history and information.

- **AtariProtos.com, www.atariprotos.com** Devoted to unearthing the secrets of Atari prototypes, AtariProtos.com features a growing repository of comprehensive prototype reviews for the 2600, 5200, and 7800. Each prototype is vigorously played and studied until enough is learned that a thorough review can be written. Even minor details between different versions of the game are highlighted and are often accompanied by many screenshots. AtariProtos.com is a great site for people who are curious about the oft-confusing world of prototypes.
- **B&C ComputerVisions, www.myatari.com** B&C ComputerVisions has been serving the Atari videogaming and Atari computing communities for nearly 20 years. B&C stocks more than 5000 Atari-related products and accepts and ships orders for Atari computers, videogames, and parts worldwide. They also service most Atari products, except monitors, power supplies, and coin-operated Atari games.
- **Best Electronics, www.best-electronics-ca.com** Best Electronics, serving Atari users for 20 years, carries a large range of replacement parts and accessories for Atari game systems and computers. Featuring one of the largest inventories of Atari parts, Best Electronics has a printed catalog of more than 200 pages with over 4000 Atari products, parts, and accessories.

Atari 5200

Topics in this Chapter:

- **Introduction**
- **Opening the Atari 5200**
- **Atari 5200 Blue LED Modification**
- **Atari 5200 Two-Port BIOS Replacement**
- **Creating an Atari 5200 Paddle Controller**
- **Freeing Yourself from the 5200 Four-Port Switchbox**
- **Atari 5200 Video and Audio Upgrade Modification**
- **Other Hacks**
- **Homebrew Game Development**
- **Atari Resources on the Web**

Introduction

In 1982, Atari released the Atari 5200 SuperSystem. Based on the Atari 400 8-bit computer, the 5200 was Atari's answer to the competition it faced in the videogame market. Atari had been working on a sequel to its 2600 VCS called the 3200, but development was halted when it was determined that the system was too difficult to write games for. Atari, knowing its 8-bit line of computers already had an excellent game library, decided instead to transform its 8-bit computer into a powerful game console.

The Atari 5200 (see Figure 9.1) was very similar to the Atari 400 computer in electrical design, but physically it was much different. The 5200 featured a sleek, wedge-shaped case with only a single power button on the unit itself. Each Atari 5200 system shipped with two analog joysticks. These new controllers allowed for a full 360 degrees of movement, but unfortunately they were not self-centering and proved awkward to use with many games. In addition, the controllers were highly prone to failure and became the primary weakness of the system. Each 5200 controller featured a numeric keypad, two independent fire buttons, and Start, Reset, and Pause buttons.

Figure 9.1 The Atari 5200 SuperSystem



Another poor design decision with the original 5200 system was the inclusion of a customized radio frequency (RF) switchbox that also routed the system's power. This unconventional approach allows the system to automatically switch the 5200 to the TV input when power is applied and eliminates the need to connect an additional cable to the 5200. Because the switchbox was a design proprietary to Atari, if your RF switchbox failed, you needed to purchase an expensive replacement from Atari. (See the "Free Yourself from the 5200 Four-Port Switchbox" hack to eliminate the need for this switchbox.)

Atari later released an updated version of the 5200 containing only two controller ports instead of the original four. The redesigned system allows the use of a standard RF switchbox, and the power supply plugs directly into the console.

Even with the Atari 5200's many shortcomings, dozens of excellent games were either ported from the Atari 8-bit computer line or developed specifically for the 5200. Many additional games were also finished or close to being completed when Atari stopped production of the system, but unfortunately these titles never saw commercial release. The Atari 5200 SuperSystem has been around for over 20 years, and in that time many hacks have been developed to help address some of the system's shortcomings as well as improve the system with various interesting modifications. The Atari 5200 also has a large community of fans and homebrew developers, all of whom are keeping the spirit of the console alive.

Let's get started!

Opening the Atari 5200

Many of the hacks in this chapter require that you open up your Atari 5200. This “hack” guides you through that fairly simple process.



Preparing for the Hack

The tools required for this hack are as follows:

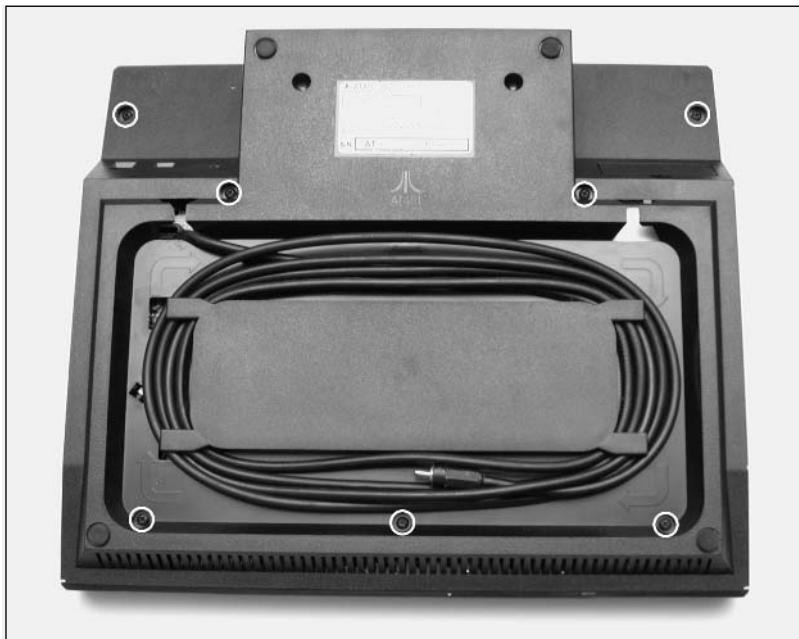
- Phillips head screwdriver, standard size
- Needlenose pliers
- Small flathead screwdriver (optional)

Before starting the hack, prepare a clean area where you can work and place all the parts that you remove from the 5200.

Performing the Hack

Perform the following:

1. First, turn the Atari 5200 upside down and remove the seven screws holding the case together. The locations of the screws are highlighted in Figure 9.2.

Figure 9.2 Atari 5200 Seven Screw Locations**WARNING: HARDWARE HARM**

The two screws at the top (as shown in Figure 9.2) are shorter than the other five screws, so you'll need to be careful to insert the longer screws into the correct holes when you reassemble the 5200. Failure to do so can result in damage to the case, especially if you insert a long screw into a hole meant for one of the shorter screws. The removed screws are shown in Figure 9.3.

Figure 9.3 Atari 5200 Case Screws

2. After you remove the screws, turn the 5200 back over and slowly remove the top of the case, exposing the printed circuit board (PCB) inside.
3. If the RF cable is plugged into the board (see Figure 9.4), simply unplug it. If the cable is soldered to the board instead (see Figure 9.5), you will have to remove the bottom half of the 5200 case to free the circuit board from the case. To do so, remove the cable from the stress-relief tabs holding the cable in place on the bottom housing, and then carefully pull the cable through the hole (see Figure 9.6).

Figure 9.4 Unplug the RF Cable (If Applicable)

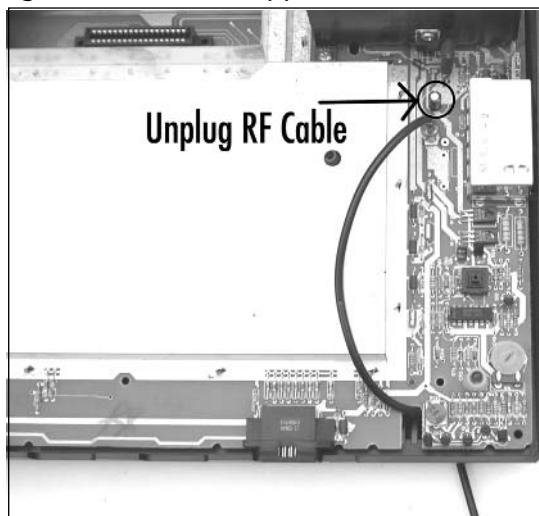


Figure 9.5 RF Cable Soldered to Board

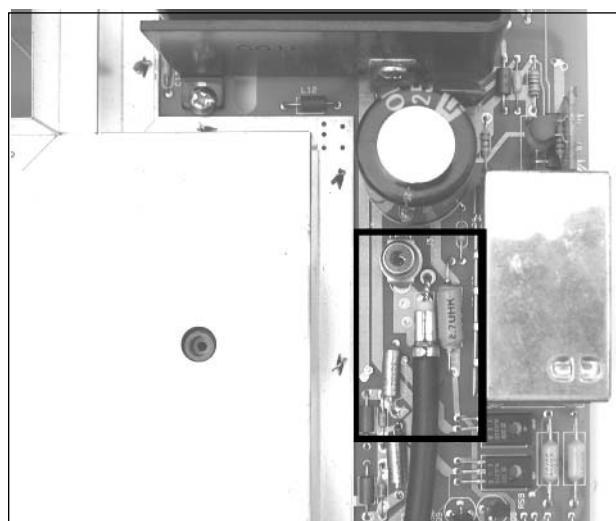
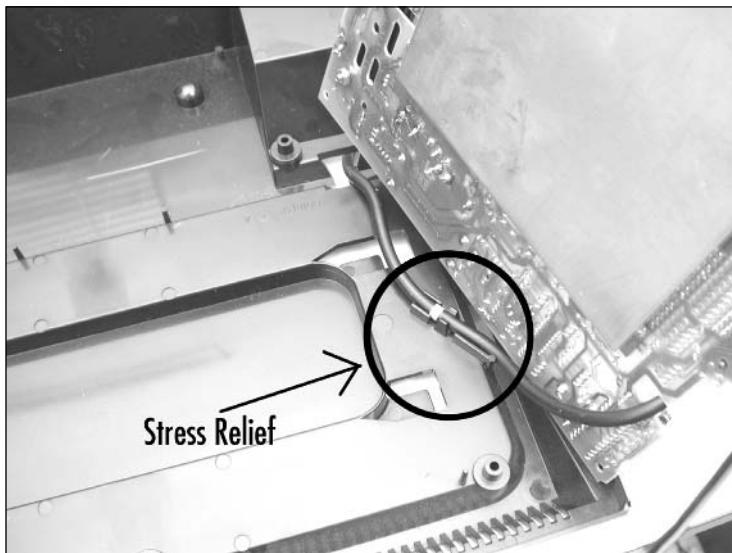
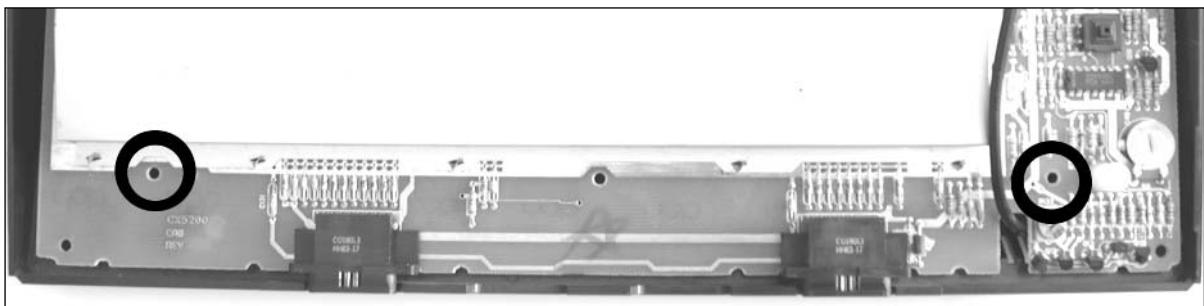


Figure 9.6 Remove the RF Cable from the Stress-Relief Tabs

Now you can remove the board from the case.

WARNING: HARDWARE HARM

Near the bottom of the board are two small plastic posts used to hold the circuit board into place. Be careful not to snap these off when you are removing the board. You can use a flat-head screwdriver to gently pry the board up between the bottom edge of the board and the case to make it easier to remove. Figure 9.7 highlights where the two plastic posts are located.

Figure 9.7 Plastic Posts to Hold the 5200 Board in Place

With the circuit board removed from the plastic case, an optional step is to remove the RF shield. The RF shield comprises two metal halves covering the majority of both sides of the 5200 circuit

board. The bottom half of the shield has metal tabs that stick through holes in the PCB and slots in the upper half of the shield. These tabs are then twisted, which holds the shield in place. Now do the following:

1. Using needlenose pliers, straighten each of the 15 tabs that secure the RF shield to the 5200 circuit board so they are in line with the slots in the top half of the RF shield (see Figures 10.8 and 10.9). It is possible that the number of tabs vary with different revisions of the Atari 5200. In order for the RF shield to be removed, these tabs must be able to fit through the slots in the top half of the RF shield, as well as through the holes in the PCB.

Figure 9.8 Five of the Fifteen RF Tabs

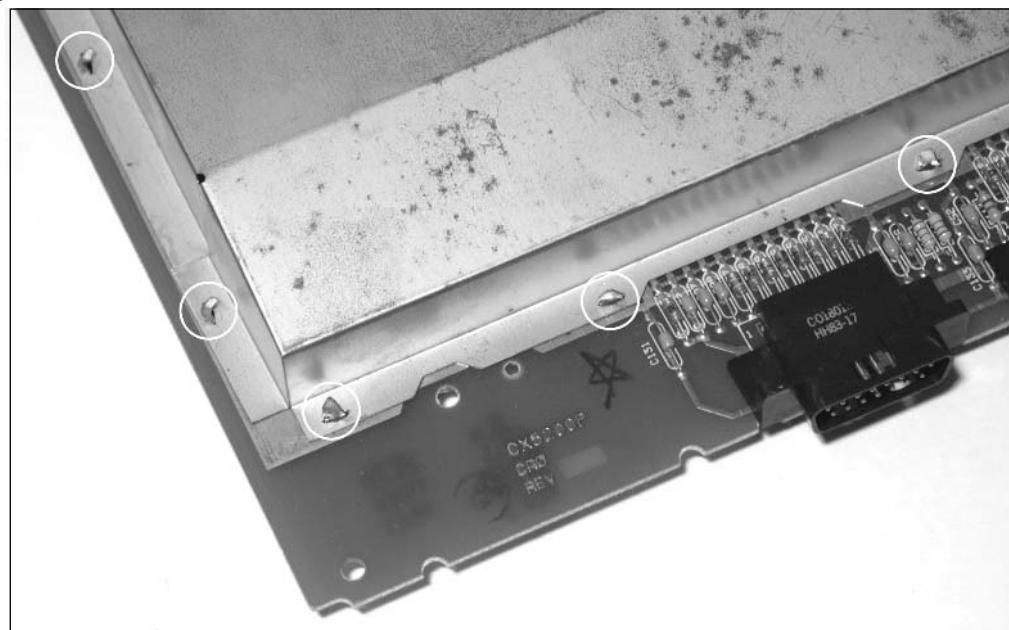
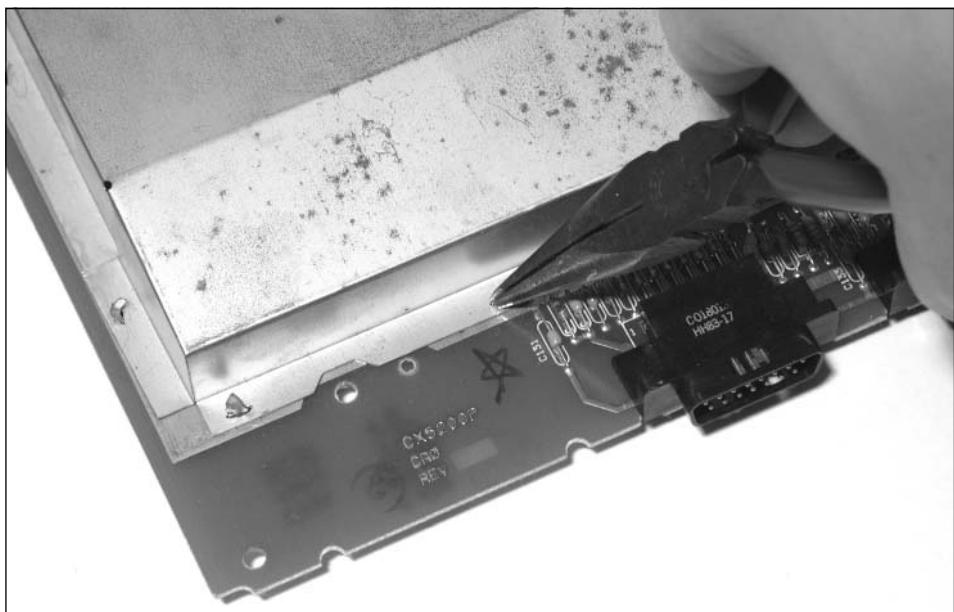


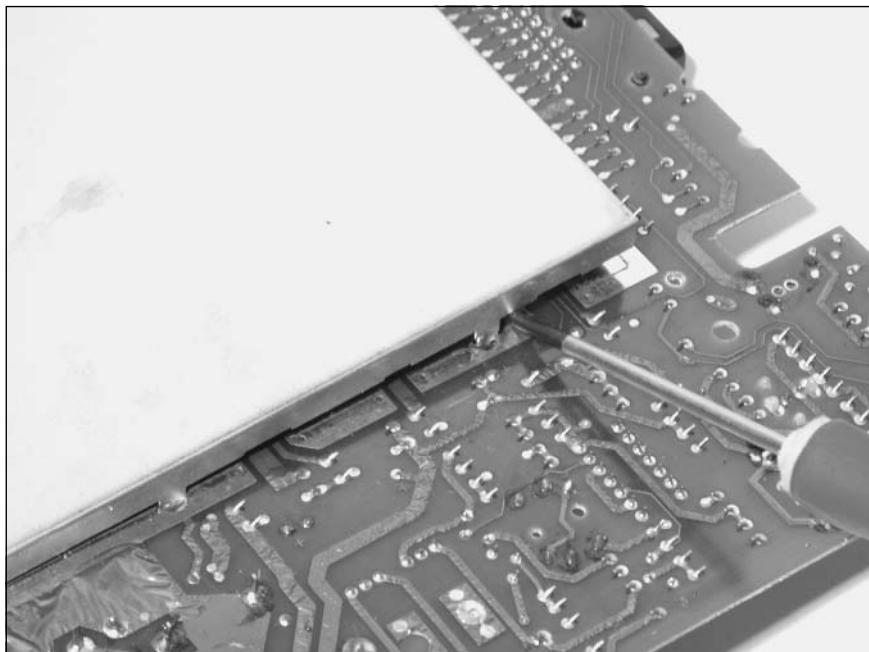
Figure 9.9 Straightening the RF Shield Tabs

2. After you have straightened all the tabs, pull the two halves of the RF shield apart (there is one on each side of the circuit board). Be careful while doing so, because the edges and corners of the RF shield are sharp and can scratch or damage the PCB. To assist with removal of the RF shield, you can use a small, flathead screwdriver to pry up the shield (see Figure 9.10).

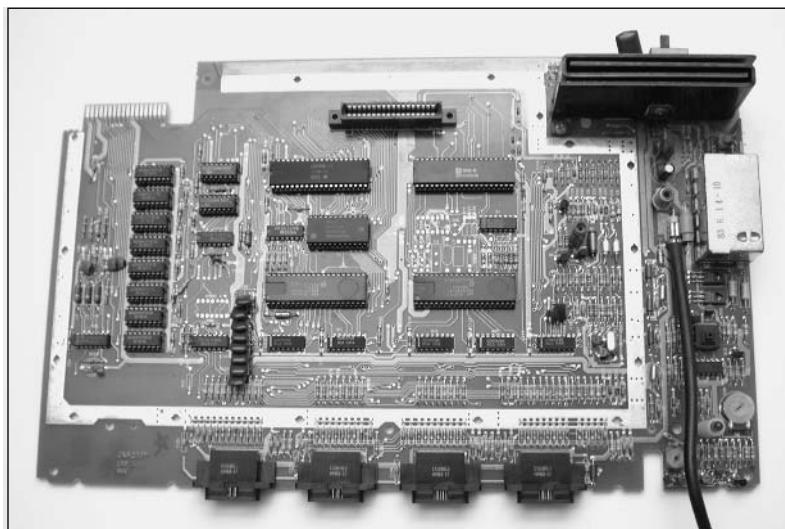
WARNING: HARDWARE HARM



If you use a screwdriver to help pry the two halves apart, take care not to place the tip of the screwdriver near any traces on the circuit board, since a single slip could cut through a trace and damage the board. Also, try not to bend the RF shield more than necessary or it might not fit back in place very well when you reassemble your 5200 at the end of the hack.

Figure 9.10 Prying Up the RF Shield

3. Once the RF shield has been removed, you can put the two halves aside until you are ready to reassemble your 5200. Figure 9.11 shows a four-port Atari 5200 board with its RF shield removed, finally free from its case.

Figure 9.11 Atari 5200 Circuit Board: Free at Last!

Reassembly

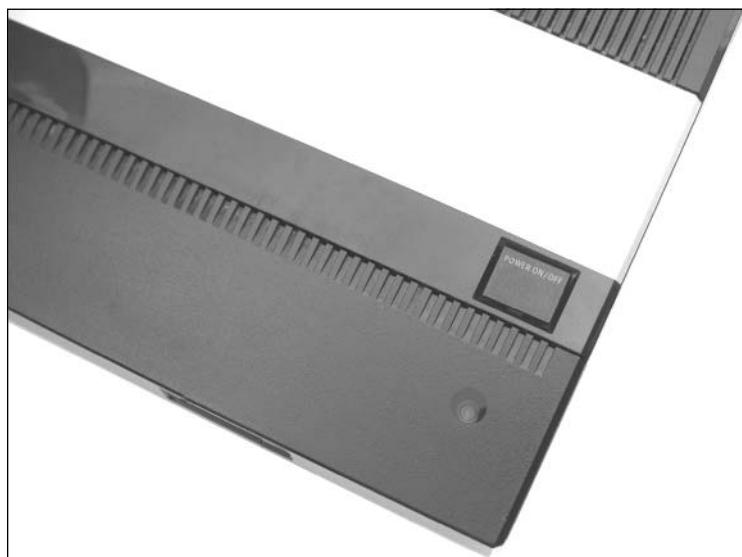
Reassembly of the Atari 5200 is straightforward—you simply follow the disassembly steps in reverse order:

1. Reattach the RF shield (if it was removed). To do this, attach the bottom half first and ensure that all the metal tabs are completely through the PCB. Then position the top half of the RF shield over the bottom half, making sure that each metal tab is placed through the appropriate slot in the top half of the RF shield. Use a pair of pliers to twist the metal tabs 90 degrees so that the tabs are perpendicular to the slots (which help hold the RF shield in place).
2. If your RF cable is soldered to your board, you'll want to feed the cable back through the hole in the bottom of the case and fasten the cable back to the stress relief (refer back to Figure 9.6).
3. Place the 5200 circuit board into the bottom of the case, taking care that the two plastic posts shown in Figure 9.7 properly fit through their respective holes in the board.
4. Reattach the RF cable to the 5200 PCB (if necessary, refer back to Figure 9.4).
5. Place the top half of the 5200 case onto the lower half.
6. While holding both halves of the 5200 case together with your hands, turn the case over and place it face down on your working surface.
7. Using a Phillips head screwdriver, fasten the case back together with the seven screws you removed at the beginning of the hack. Remember that the two short screws are for the top two holes. If you use the long screws in these holes, you could damage your case.

That's it! Now you can enjoy the hack (or hacks) you just completed!

Atari 5200 Blue LED Modification

The Atari 5200 uses a momentary push-button switch to toggle power on and off to the system. Since the switch doesn't give a visual indication of whether the unit has power or not, Atari added a red LED that illuminates when the unit is on (see Figure 9.12). This hack explains how you can replace this stock LED with a blue LED to add a unique touch to your Atari 5200.

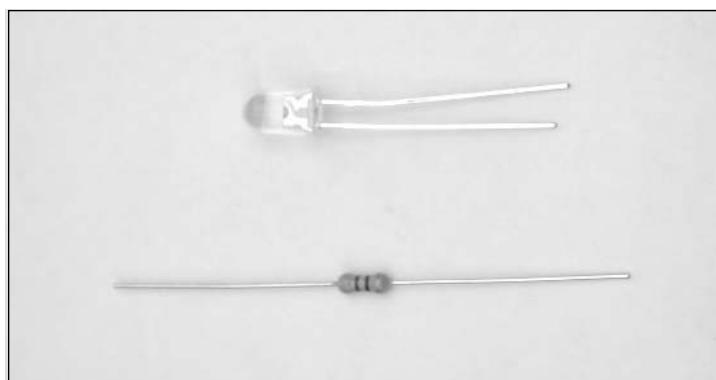
Figure 9.12 Atari 5200 with the Standard Red LED

Preparing for the Hack

For this hack you'll need two components (see Figure 9.13):

- A blue LED (2600mcd, 3.7V, 20mA, Radio Shack part #276-316)
- A 470 ohm, 5% resistor (Radio Shack part #271-1317)

We'll use a blue LED with a forward voltage of 3.7V and a brightness of 2600mcd at 20mA, coupled with a 470 ohm current-limiting resistor. You can experiment with different values of resistance if you'd like a brighter or dimmer LED. The resistor installed in the 5200 is 150 ohms, which, if paired with the new blue LED, results in significantly higher light output.

Figure 9.13 Required Parts

The tools required for this hack are:

- Phillips head screwdriver, standard size
- Soldering iron
- Solder sucker or solder braid
- Wire cutters

Performing the Hack

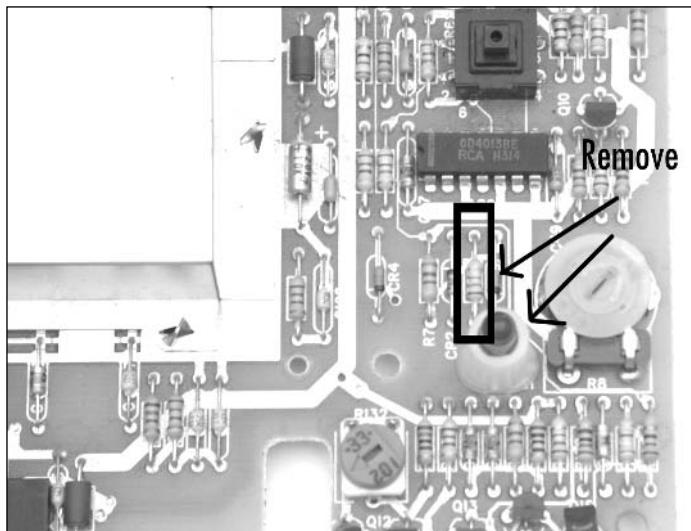
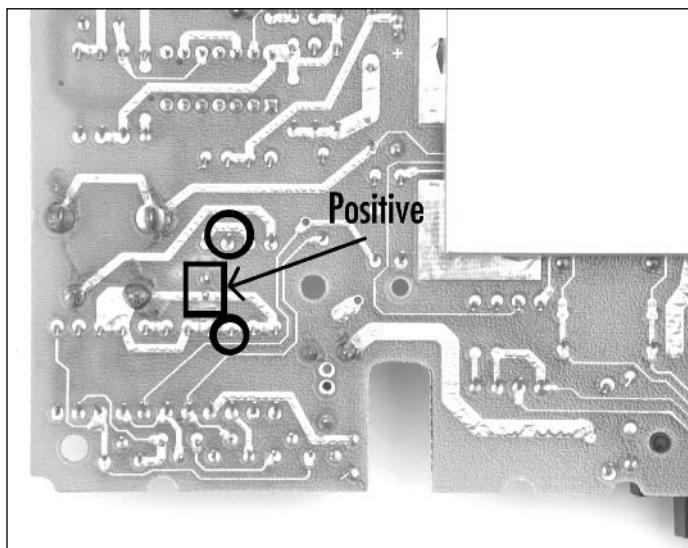
Perform the following:

1. First, you need to open your Atari 5200 as described in the “Opening the Atari 5200” section at the beginning of this chapter. You do not need to remove the RF shield for this hack.
2. We’re going to be working with the lower-left portion of the circuit board, where the LED is located. First, unsolder the existing LED and the accompanying resistor (see Figures 9.14 and 9.15). To remove the LED from the board, you will need to squeeze the two plastic tabs holding the plastic housing for the LED to the board.

WARNING: HARDWARE HARM



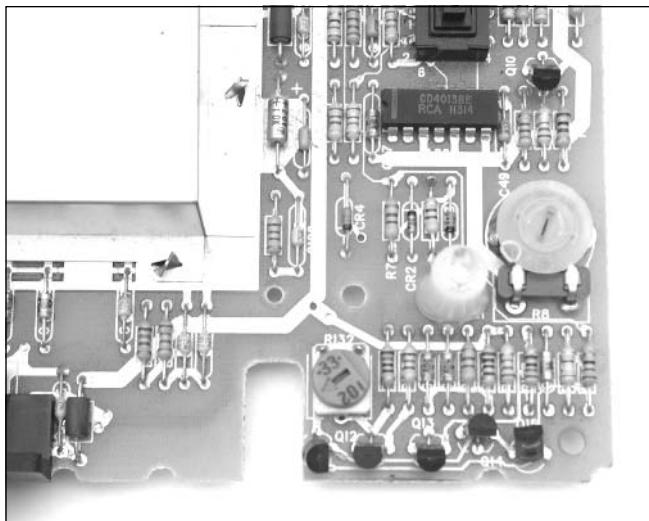
Be careful when removing the LED from the board; you don’t want to pull up the solder pads from the front side of the board, which you don’t have access to due to the LED’s plastic spacer. Try to remove as much solder as possible from the joints on the bottom of the board, and then carefully pull the LED up from the board while you heat the connections.

Figure 9.14 Resistor and LED**Figure 9.15** Unsolder the Resistor and LED Connections

3. Once you have unsoldered the resistor and LED, remove them from the board.
4. Remove the LED from the white plastic spacer.
5. Place the blue LED into the white plastic spacer and then solder it to the board. The long lead of the LED is the anode (positive) and must be soldered to the upper connection, as shown in Figure 9.15.

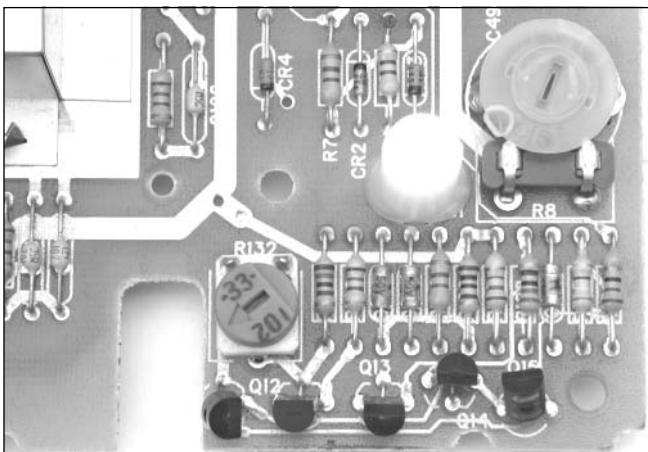
6. Next, solder the 470 ohm resistor into the space formally occupied by the original resistor you removed.
7. Use the wire cutters to cut any excess from the resistor and LED leads after you've soldered them in. When you're done, the LED and resistor should resemble Figure 9.16.

Figure 9.16 New LED and Resistor



8. If you don't have anything underneath the 5200 that could short the connections on the bottom of the circuit board, you can plug in the power supply and press the power button (the square, black button above the LED) to test your hack and to see how the new LED looks (see Figure 9.17). However, you won't have an accurate feel for how the LED looks until you put the 5200 back together and see it shining through the dark plastic of the case.

Figure 9.17 Power Applied



9. You can now reassemble your 5200 as described in the previous “Opening the Atari 5200” section. Plug your 5200 back in, insert a cartridge, and power it up. Enjoy the new look of your Atari 5200 (see Figure 9.18)!

Figure 9.18 Blue LED Happiness on the Atari 5200



Under the Hood: How the Hack Works

The “Blue Power LED Modification” hack in the Nintendo NES chapter and the “Blue LED Modification” hack in the Atari 7800 chapter both feature details of how the hack works and why the particular LED and resistor values were chosen. The only difference with the 5200 is that an unmodified Atari 5200 uses a 150 ohm resistor to limit the current flow through the LED, compared to a 220 ohm resistor found in the NES and a 120 ohm resistor found in the 7800.

Atari 5200 Two-Port BIOS Replacement

There are two primary versions of the Atari 5200. The original model of the Atari 5200 has four controller ports and requires the use of a proprietary RF switchbox into which the power supply is plugged. Atari later released a version of the 5200 with only two controller ports and which does not require the proprietary switchbox. In addition to these obvious changes, Atari also changed the Basic Input/Output System (BIOS) code built into the 5200. The BIOS contains various low-level routines that allow programs written for the 5200 to access specific hardware in the console. When Atari changed the BIOS for its two-port 5200 version, the company accidentally broke compatibility with three third-party games:

- Pitfall, by Activision
- Mountain King, by CBS Electronics
- K-Razy Shootout, by CBS Electronics

Fortunately, it is relatively easy to replace the BIOS in the two-port Atari 5200 with the BIOS from a four-port Atari 5200, which will re-enable compatibility with the three games.



Preparing for the Hack

For this hack, you'll need two components:

- A two-port Atari 5200 system
- A four-port BIOS chip

You can obtain a four-port BIOS chip from a nonworking Atari 5200 system, which is what we'll use in this hack. Alternatively, a more advanced option is to program the BIOS code (available at www.atariage.com/5200/roms/5200.zip) into a 2532 Erasable Programmable Read-Only Memory (EPROM) using a device programmer.

The tools required for this hack are:

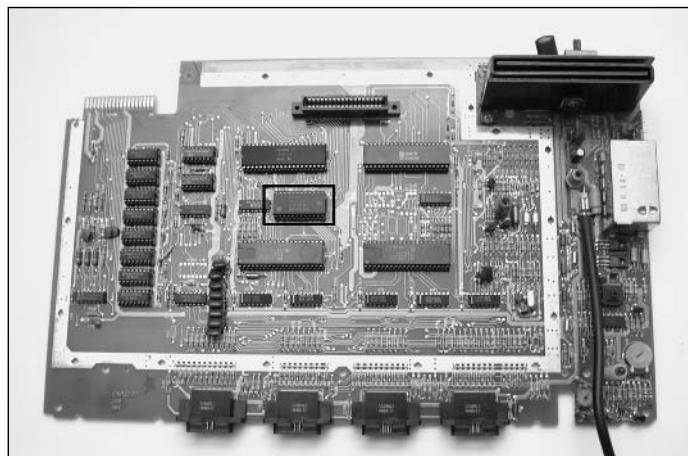
- Phillips head screwdriver, standard size
- Needlenose pliers
- Small flathead screwdriver
- An integrated circuit (IC) extraction tool (optional)

Performing the Hack

Perform the following:

1. Open the four-port Atari 5200 as described in the “Opening the Atari 5200” section earlier in this chapter. This hack requires that you remove the RF shield. Your board should resemble the one shown in Figure 9.19.

Figure 9.19 Atari 5200 Four-Port Circuit Board



WARNING: HARDWARE HARM



Be sure to take proper antistatic precautions before working with electronic circuitry. Integrated circuits are extremely susceptible to static discharge and can be destroyed if they are not handled properly. All electronics should be handled at a static-safe workstation with electrostatic discharge (ESD) mats and grounded wrist and/or ankle straps.

2. Remove the BIOS chip from the circuit board (see Figure 9.20). Using a small flathead screwdriver, carefully pry between the chip and the socket to loosen the chip. Work from one end, then switch to the other side, and repeat as necessary to work the chip out of the socket (see Figure 9.21). Be careful not to bend any of the pins while you are doing so. If you are using an IC extraction tool, simply pull the device straight up and out of the socket.

Figure 9.20 Atari 5200 Four-Port Circuit Board Showing the BIOS

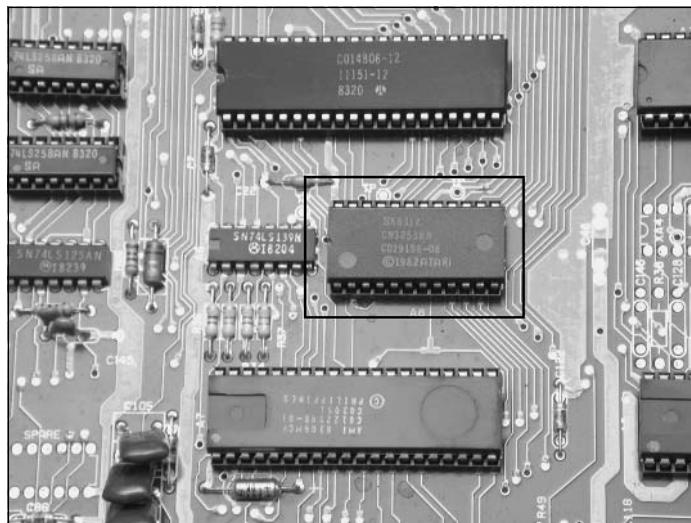


Figure 9.21 Removing the Four-Port BIOS

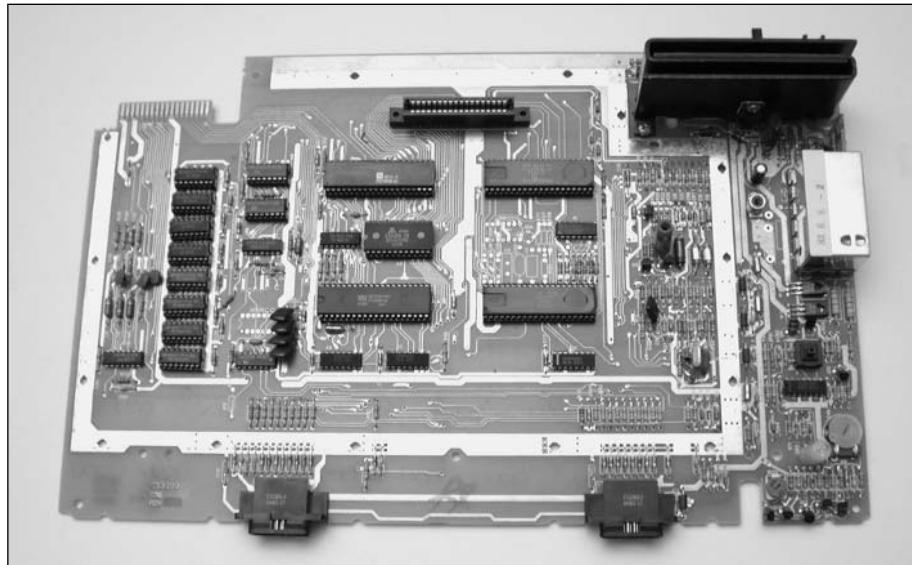
3. After you have removed the BIOS chip (shown in Figure 9.22), set it safely aside for now. The part number for the four-port BIOS chip is C019156 (the digits after the dash are not important). The newer two-port BIOS part number (the BIOS with the compatibility problems that we are replacing in this hack) is C019156A.

Figure 9.22 Close-Up of the Four-Port BIOS

You can now put aside the four-port system, since we no longer need it for the remainder of this hack. The next part of the hack is to open the two-port system, remove the original BIOS, and replace it with the one that we obtained from the four-port system.

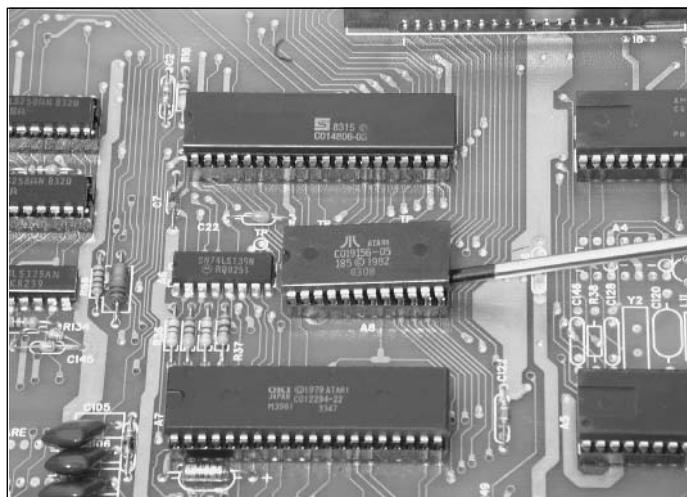
4. Open the two-port Atari 5200 as described in the “Opening the Atari 5200” section earlier in this chapter. When the circuit board is removed from the case, it should resemble the one shown in Figure 9.23.

Figure 9.23 Atari 5200 Two-Port Circuit Board



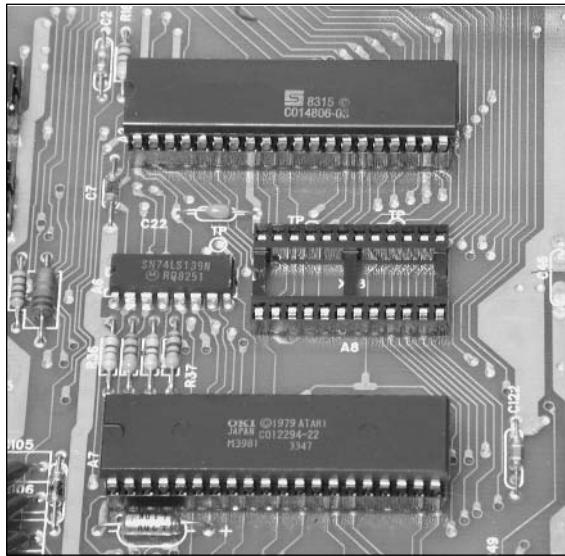
5. The BIOS chip in the two-port Atari 5200 is in the same location as in the four-port system. Using a flathead screwdriver or IC extraction tool, carefully remove the BIOS from the two-port board (see Figure 9.24).

Figure 9.24 Removing the Two-Port BIOS



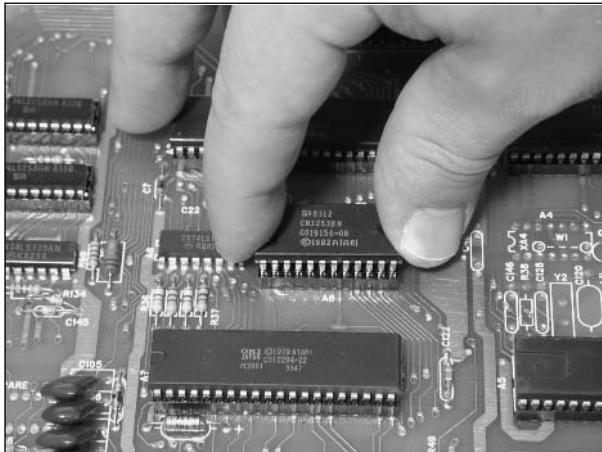
6. After you've removed the BIOS, your two-port board should resemble the one shown in Figure 9.25.

Figure 9.25 Two-Port PCB with Empty BIOS Socket



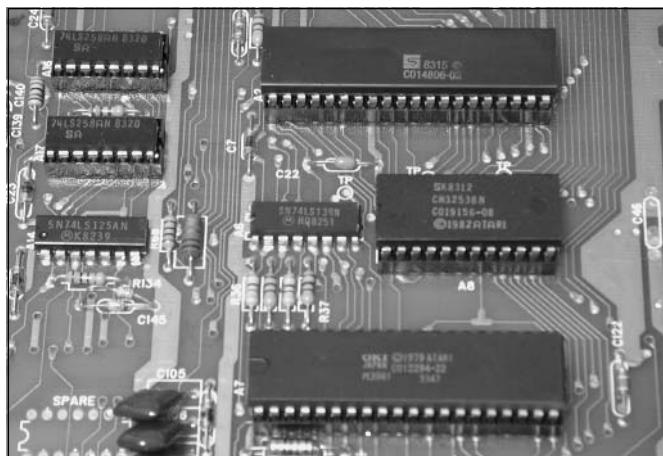
7. Now it's time to place the four-port BIOS chip into the socket formerly occupied by the two-port BIOS. Before inserting the chip, make sure that the notch on the top of the chip is aligned with the notch on the socket. With your board aligned as shown in Figure 9.26, the notch will face to the left (the same direction as all the other chips in the picture). Place the chip on the socket, making sure that all the pins are resting in the holes of the socket. Once you are comfortable that all the chips are properly aligned, apply pressure to the chip and push it straight down into the socket (see Figure 9.26).

Figure 9.26 Replacing the BIOS



- After you have inserted the chip, inspect it to make sure that all the pins are properly seated in the socket. Your board should resemble the one shown in Figure 9.27.

Figure 9.27 Successful Replacement of the BIOS



- Finally, reassemble your 5200 as described in the previous “Opening the Atari 5200” section. Now you can enjoy the entire Atari 5200 library of games on your two-port console!

Creating an Atari 5200 Paddle Controller

Unlike the Atari 2600, with which Atari included a pair of paddle controllers (see Figure 9.28), the Atari 5200 shipped with analog joystick controllers (see Figure 9.29). Fortunately for hardware hackers, the Atari 5200 controller can be merged with an Atari 2600 paddle, allowing you to enjoy a few select games that work very well with it. These games (see Figure 9.30) include:

- Gorf
- Kaboom!
- Moon Patrol
- Super Breakout

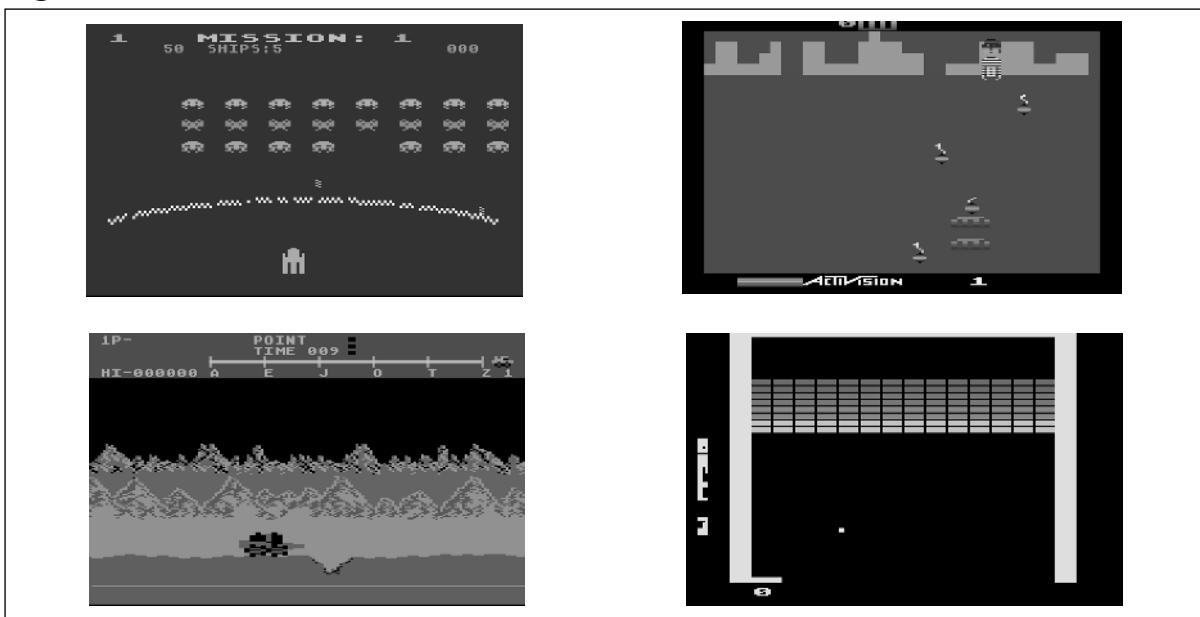
Figure 9.28 An Atari 2600 Paddle Controller



Figure 9.29 An Atari 5200 Controller



Figure 9.30 Atari 5200 Paddle-Based Games



Additionally, a new homebrew game for the 5200, Castle Crisis, has been developed with direct support for the Atari 5200 paddles described in this hack. Other games will also work with the paddle controller, notably any games that require only horizontal movement. However, the preceding list highlights those games that work best with it. Once you see the final result of this hack, you'll be amazed at how the 5200 controller appears to have been designed with the 2600 paddle controller in mind—the two are a near-perfect match!



Preparing for the Hack

For this hack, you'll need the following components:

- An Atari 5200 controller
- An Atari 2600 paddle controller
- Eight 3/8-inch hex nuts (optional)

The two most important items are the Atari 5200 controller and the Atari 2600 paddle controller, both of which you will hack apart to create the new 5200 paddle controller. You'll want to ensure that you find a 5200 controller that has good working fire buttons and keypad—the condition of the joystick itself is unimportant since we'll be replacing it with the rotary mechanism from the 2600 paddle controller. As for the 2600 paddle controller, try to find one that is free from the jitter that often plagues old 2600 paddle controllers, because this jitter will translate to the 5200 controller, making your new controller less enjoyable to use. (However, this jitter can be fixed; see the “Revitalize Your Atari 2600 Paddles” hack in the Atari 2600 chapter for more information.)

As an optional part of this hack, you can also weigh down the paddle controller using 3/8-inch hex nuts to give it a higher-quality, heavier feeling that you might find in an arcade rotary controller. If you do this part of the mod, you'll need to use hot glue to secure the nuts into the base of the spinner.

The tools required for this hack are:

- Phillips head screwdriver, standard size
- Needlenose pliers
- Wire cutters
- Super Glue
- Soldering iron (optional)
- Hot-glue gun (optional)

Performing the Hack

The creation of the Atari 5200 paddle controller is a multistep process. To make this process easier to understand, the hack is divided into sections, as follows:

- Disassembling the Atari 2600 Paddle Controller
- Building the 5200 Paddle Controller
- Adding a Weighted Dial

Disassembling the Atari 2600 Paddle Controller

The first task that needs to be completed is to disassemble the Atari 2600 paddle controller. Perform the following:

1. First, pull the dial off of the Atari 2600 paddle controller. It should pop off easily; just make sure you pull straight upward (see Figure 9.31).

Figure 9.31 Removing the Paddle Dial



2. Using the needlenose pliers, carefully unscrew the nut that was underneath the dial (see Figure 9.32).

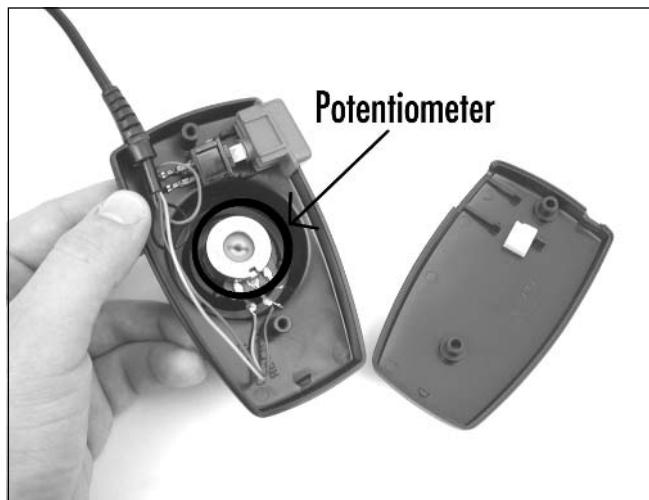
Figure 9.32 Removing the Paddle Nut



3. Flip the paddle controller over and remove the two screws with a Phillips head screwdriver.

4. Separate the two halves of the controller (see Figure 9.33).
5. Pull out the paddle mechanism (the potentiometer, also known as a *pot*, as denoted in Figure 9.33) and disconnect the two wires plugged into it. If the wires are soldered to the potentiometer, you'll need to first desolder them. You'll want to do this cleanly because we'll be attaching two wires from the 5200 controller when we transplant the potentiometer.

Figure 9.33 Disassembled Paddle



Now that you have the paddle controller disassembled, you'll want to put aside the potentiometer, nut, and plastic dial (see Figure 9.34).

Figure 9.34 Save These Paddle Parts



Building the 5200 Paddle Controller

With the paddle disassembly out of the way, we can now concentrate on the creation of the 5200 paddle controller. Perform the following:

1. Remove the three screws from the bottom of the 5200 controller (see Figure 9.35).

Figure 9.35 Removing the Three Screws from the Back of the 5200 Controller



2. Before you pull the controller apart, you first need to pop the bezel off the top of the controller; the bezel houses the Start, Pause, and Reset buttons. Using a thin flathead jeweler's screwdriver, carefully pry between the bottom of the bezel and the 5200 controller to slowly raise it up (see Figure 9.36). It should eventually pop out, along with the rubber buttons underneath. Place all the removed pieces aside for now.

Figure 9.36 Removing the Upper Bezel



3. Under the rubber buttons are traces mounted on a piece of thin, flexible plastic. This is part of a much larger sheet that contains the traces for all the buttons in the controller (see Figure 9.37). Carefully slide this out from a slot on the left side of the controller.

WARNING: HARDWARE HARM



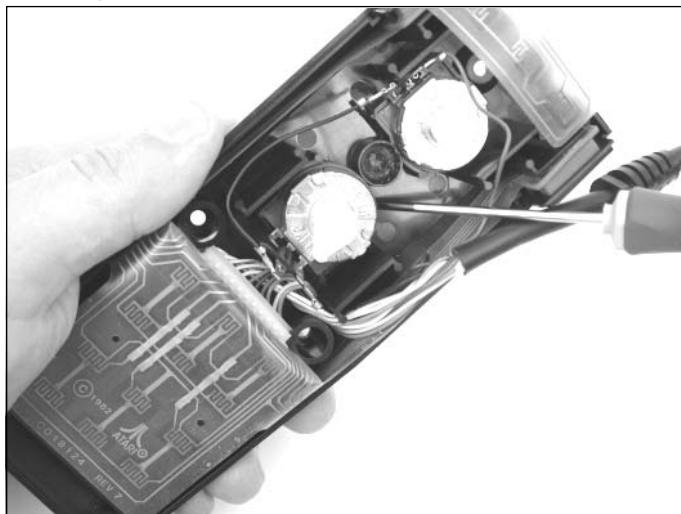
The 5200 controller contains a single, folded Mylar sheet with all the traces for the keypad buttons, fire buttons, and Start, Reset, and Pause buttons. Care must be taken not to scratch or damage this Mylar or you could render your 5200 controller unusable.

Figure 9.37 Bottom Half of the Atari 5200 Controller



4. Now pull the two halves of the controller apart. There are two posts at the bottom of the controller, and you will need to lift the top half straight up to get the controller apart. Place the top half of the controller (containing the joystick mechanism) aside for now.
5. The bottom of the controller contains two potentiometers—one that controls the horizontal movement and one that controls the vertical movement. Both of these pots need to be removed, but they are glued to the base of the controller. The easiest way to remove them is to insert a thin, flathead screwdriver under the base of the mechanism and twist the screwdriver (see Figure 9.38). As the glue separates, you should hear it peeling, and the pots will soon be freed.

Figure 9.38 Removing the Two Potentiometers



- Once you have removed the potentiometers from the base of the controller, you can disconnect the wires from the upper potentiometer (see Figure 9.39).

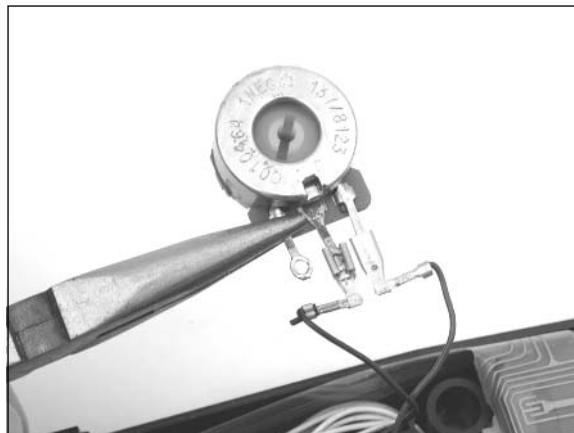
Figure 9.39 Removing Wires from the Upper 5200 Potentiometer



- Cut the wires that you just removed from the upper potentiometer as close to the cable jacket as possible. We won't be needing these connections anymore, so getting rid of them will give us more room to work inside the controller housing.

8. Shift your attention to the bottom potentiometer. Remove the middle connector and plug it into the middle connector of the potentiometer you removed earlier from the 2600 paddle controller.
9. Remove the other wire from the same 5200 pot and, while holding the 2600 pot with the plastic post face down and the connectors pointed toward you, attach this wire to the right-most connector (see Figure 9.40). You might be able to slide the connectors on, but if they don't fit you'll have to solder them into place.

Figure 9.40 Wiring the 2600 Potentiometer in Place of the Bottom 5200 Potentiometer



We're done with the base—now we need to spend time on the upper half of the controller, which contains the joystick mechanism.

10. We need to remove the joystick that is still attached to the upper half of the controller. A small, white, rectangular ball joint prevents the joystick from coming out. We'll need to cut this housing along one edge so that the joystick can then be pulled straight out. To do this, use a pair of wire cutters, a sharp knife, or a Dremel tool to cut through one side of the white plastic housing (see Figure 9.41). Take care not to damage the two sliding plates (one is black, the other white) underneath this housing, because we need those to complete the hack.

Figure 9.41 Freeing the Joystick

11. Once you have successfully cut and removed the ball joint, pull the joystick out from the top side of the controller housing. Leave the rubber boot in place for now.
12. Place the top half of the controller housing upside down on a flat surface. Remove the top white plastic plate and slide the bottom plate so that the hole in the middle is aligned with the hole of the rubber boot. Using Super Glue, glue the edges of this plate into place.
13. After waiting for the glue to dry, place the white plate back into its tray and align it, too, so its window is centered with the hole of the rubber boot. Super Glue it into place. The two plates should resemble Figure 9.42 when you are done.

NOTE

Take care to properly align these plates, because you will be placing the post of the 2600 paddle controller through these two openings. If they are not properly aligned, the paddle dial will be off-center in relation to the round edge on the exterior of the 5200 controller.

Figure 9.42 Glue the Plastic Sliding Plates into Place

14. After waiting for the glue to dry again, flip the top half of the controller over so it is right-side up. Using a pair of pliers, remove the rubber joystick boot (see Figure 9.43).

Figure 9.43 Removing the Rubber Joystick Boot

15. Now take the 2600 paddle mechanism and place the post of the potentiometer through the plates so it protrudes through the top half of the 5200 controller. Align the connectors of the potentiometer with the bottom of the controller.
16. Use the hex nut you saved from disassembling the 2600 paddle controller and thread it onto the 2600 potentiometer (see Figure 9.44). Tighten the nut, making sure that the paddle mechanism is properly aligned on the back side of the controller housing (see Figure 9.45).

Figure 9.44 Inserting the 2600 Potentiometer



Figure 9.45 Inserting the 2600 Potentiometer, Inside View



17. You can now reassemble the controller (see Figure 9.46). The easiest way to do this is to hold the top half of the controller upside down, place the rubber keypad buttons into the keypad holes, and then slide the thin Mylar for the Start, Pause, and Reset buttons through the slot at the top of the controller.

Figure 9.46 Reassembling the Controller

18. Carefully fit the two halves of the controller together, making sure not to pinch the Mylar film at the top of the controller. Once the two halves are snug, you can then snap the Start, Pause, and Reset buttons and their bezel into the top of the controller.
19. Finally, place the paddle dial on top of the potentiometer shaft. Before replacing the three screws that hold the controller together, you should first test the controller with a paddle-supported Atari 5200 game (Gorf, Kaboom!, Moon Patrol, or Breakout) to make sure that it works properly.

Congratulations! You now have a 5200 paddle controller that looks as though it came straight from the Atari factory (see Figure 9.47). If you want to improve the controller even further, read on!

Figure 9.47 The Atari 5200 Paddle Controller

Adding a Weighted Dial

An optional step in the creation of your Atari 5200 paddle controller is to weigh down the dial to give it a more realistic arcade feel. For this simple step, you will need eight 3/8-inch hex nuts and a hot glue gun.

Perform the following:

1. Start by popping off the plastic rotary dial from the top of the 5200 paddle controller. Flip the dial upside down and stack two hex nuts into each compartment of the dial, for a total of eight nuts (see Figure 9.48).

Figure 9.48 Weighing Down the Dial with Hex Nuts



NOTE



Instead of using two 3/8-inch hex nuts in each compartment, you can substitute six pennies standing vertically.

2. Using the hot-glue gun, fill the dial with a sufficient amount of glue to keep the nuts in place (see Figure 9.49). Fill the center of the nuts and then the gaps around the sides of the nuts between the edges of each compartment. Try not to get glue on the top outer edge of the dial or in the center hole where the dial attaches to the potentiometer.

Figure 9.49 Gluing Your Nuts into Place

3. After the glue has had time to set, reattach the dial to the controller.

You now have a weighted 5200 paddle controller that feels as though it belongs in an arcade!

Under the Hood: How the Hack Works

Although the Atari 8-bit computers (on which the Atari 5200 is based) use standard, digital controllers with four directions (and diagonals), Atari decided to design unique and “innovative” analog controllers for its 5200 console. These controllers allow 360 degrees of rotation and variable movement in any direction using two potentiometers.

A *potentiometer* is a variable resistor for which the output is determined by the position of a “sweeper” that moves along a resistive strip. The resistance for each potentiometer in the 5200 controller ranges from 0 ohms to 500k ohms, with 250k ohms when the potentiometer is in the center position. As the sweeper moves, the resistance changes in a linear fashion, which is then read by the 5200 and converted to a discrete value that the game software can read. One potentiometer is used for horizontal position and another is used for vertical position. Fortunately for us, the potentiometers in the 5200 controller and the 2600 paddle controller can easily be swapped because they are the same physical size, though the values are different. Since we’re only interested in horizontal movement, our modified 5200 controller only needs a single potentiometer.

This hack is made easier for us due to the fact that the 5200 controller housing seems to have been designed with the potential for an official Atari 5200 paddle controller, so retrofitting the potentiometer from the Atari 2600 paddle is a piece of cake. What appears to be an actual Atari 5200 paddle controller (Model #CX-52P, versus Model #CX-52 for the standard Atari 5200 controller) has been discovered in recent years, but it’s unknown whether this was indeed an official Atari prototype or a controller that someone modified in a fashion similar to the hack described in this chapter.

Freeing Yourself from the 5200 Four-Port Switchbox

When Atari designed the Atari 5200, the company decided to create a unique, proprietary TV/game switchbox that would not only supply the audio/video signal to the television but also provide power to the four-port 5200. The Atari 5200 power supply is plugged directly into the switchbox instead of into the 5200, thus reducing cable clutter. Additionally, when power is applied to the 5200, the switchbox automatically switches the video/audio signal from the 5200 to the antenna input of the TV, saving the user from having to get up and manually slide a switch, as was required on most other TV/game switchboxes of the time (see Figure 9.50).

Figure 9.50 The Common TV/Game Switchbox

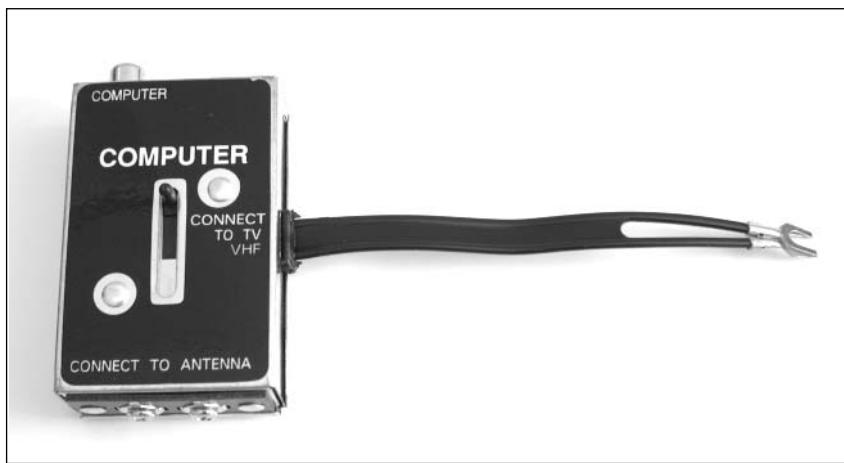


Figure 9.51 shows the switchbox that shipped with the 5200. The RCA connector at the upper-right corner plugs into the 5200. The connector at the bottom (near the sliding switch) is where the power supply is connected.

Figure 9.51 The Atari 5200 Four-Port Switchbox



Although Atari's intentions were noble, this proprietary switchbox precluded the use of a standard switchbox, since it was the only way to supply power to the 5200. This wasn't a problem when Atari was still producing the 5200, because replacement switchboxes were easy to come by, though more expensive than a standard switchbox. However, the most common problem today is finding a switchbox to go with your four-port 5200. Atari recognized the problems of this switchbox and ultimately resorted to a standard RF switchbox and separate power supply connector when it later introduced the two-port Atari 5200.

The hack in this section removes the four-port Atari 5200's dependence on this special switchbox, allowing you to use a normal TV/game switchbox. As part of this hack, we'll add a connector to the back of your 5200 where the power supply will be plugged directly into the console.

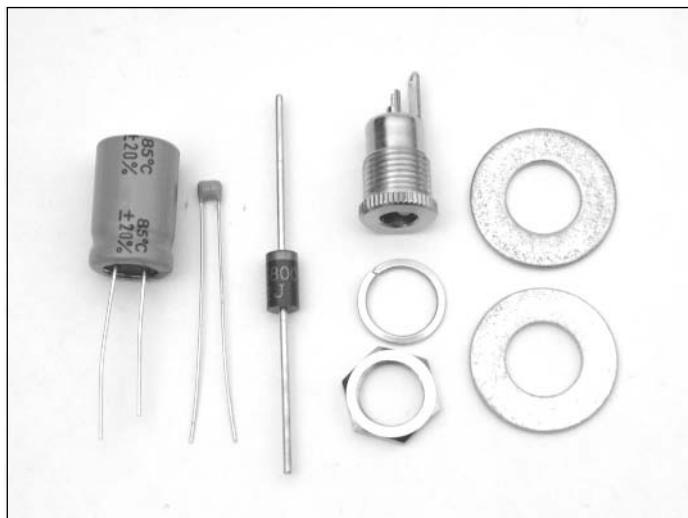


Preparing for the Hack

This hack requires the following parts:

- Size N power jack, 2.5-inch inner diameter, 5.5-inch outer diameter (Radio Shack part #274-1576)
- 47uF electrolytic capacitor, 35V or higher (Radio Shack part #272-1027)
- 0.1uF ceramic capacitor, 50V (Radio Shack part #272-109)
- 1N5391G rectifier diode (this part is difficult to obtain, but the NTE5800 is a close match and is what we will use to perform the hack; you can also substitute a 1N4001 diode if you cannot find the others)
- Metal washers (two), 7/16-inch inner diameter
- 18AWG wire (two pieces, each about a foot long, preferably red and black)
- 3/16-inch diameter heat-shrink tubing
- 3/32-inch diameter heat-shrink tubing
- Electrical tape

To mount the power connector to the 5200 case, you'll need the two 7/16-inch metal washers to facilitate fitting the connector in the already existing large rectangular hole at the back of the 5200 case. This existing hole was actually used in the two-port version of the 5200 to expose its built-in power connector. If you prefer, you can simply drill a 7/16-inch hole in the back of the 5200 case, rendering the washers unnecessary and resulting in a slightly cleaner appearance. Some 3/16- and 3/32-inch heat-shrink tubing will also be used to shield the connections on the power cable we'll be building. All the required parts are shown in Figure 9.52.

Figure 9.52 Required Parts

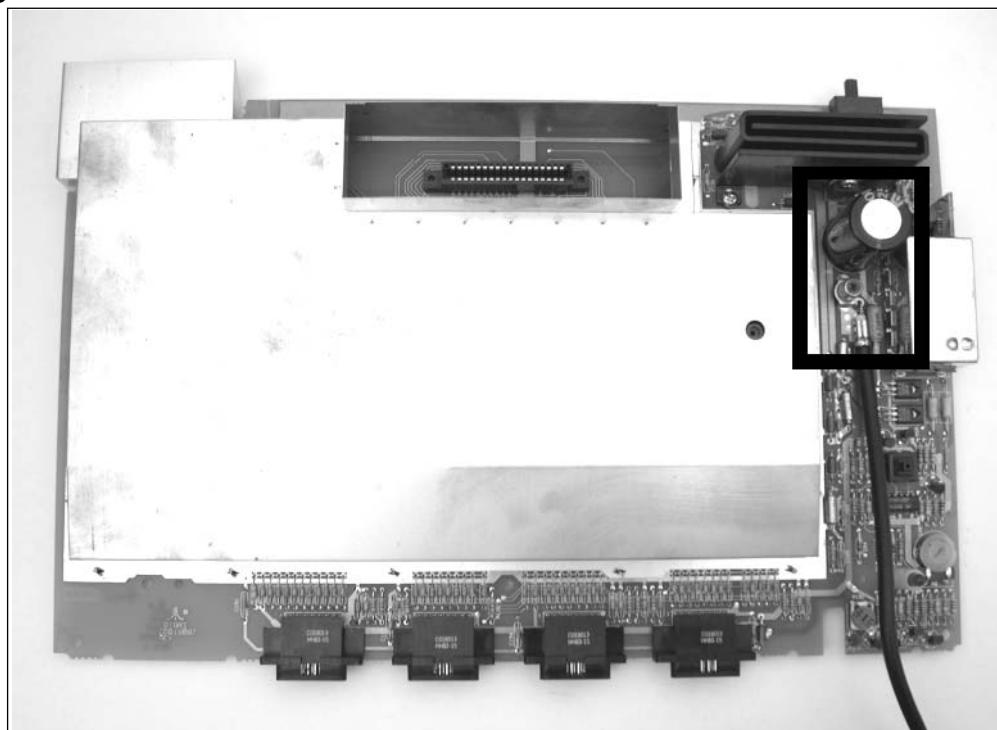
The tools required for this hack are:

- Phillips head screwdriver, standard size
- Wire cutters
- Wire stripper
- Soldering iron
- Small adjustable wrench or needlenose pliers
- Heat gun

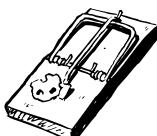
Performing the Hack

Perform the following:

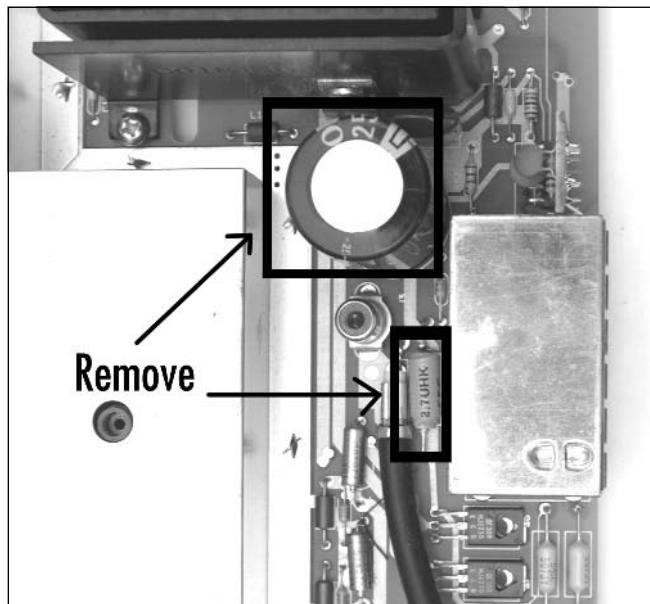
1. Open the four-port Atari 5200 as described in the “Opening the Atari 5200” section earlier in this chapter. You do not need to remove the RF shield for this hack. Your board should resemble the one shown in Figure 9.53, which also highlights the area of the PCB we’ll be working on (in the upper-right corner).

Figure 9.53 The 5200 Four-Port Circuit Board

2. Next, two components need to be removed from the board: L8, a 2.7uH inductor located immediately to the left of the metal RF box, and C45, a large 4700uF electrolytic capacitor located just above the video input connector. Both components are denoted in Figure 9.54.

WARNING: HARDWARE HARM

Care must be taken in removing the capacitor from the 5200. Because of the capacitor's large size and location on the board, you will only have access to the solder joints on the bottom side of the board. Remove as much solder as you can from the bottom of these joints, and slowly work out each of the capacitor's leads by heating the joint and pulling up on the capacitor. Failure to do this carefully may result in traces being lifted from the front side of the circuit board when you pull up the capacitor, thus damaging your system.

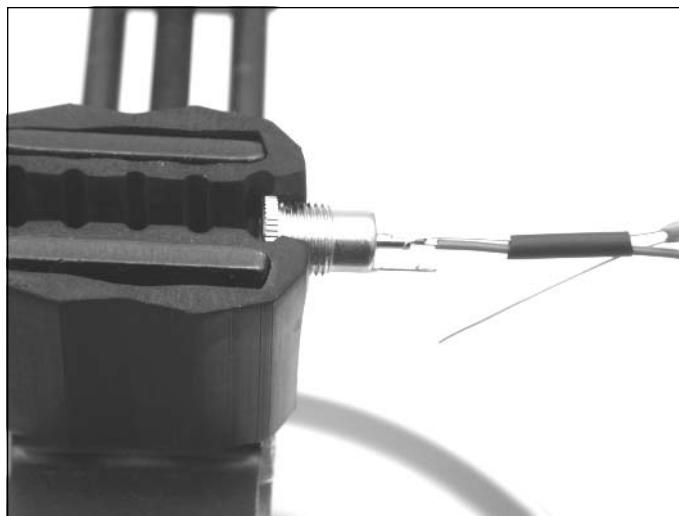
Figure 9.54 Remove the Two Components

3. With these components removed, it's time to build a cable that will connect the 5200 circuit board with the power connector that we'll be attaching to the case. Start by cutting two lengths of the 18-gauge wire, each about a foot long. If possible, use red and black wires (red for positive, black for ground).
4. Strip a short piece of insulation from each end of the red wire. Solder the anode end (the side without the stripe) of the 1N5391G diode to one side of the wire. After you've soldered the diode to the wire, use a piece of small-diameter heat-shrink tubing to cover the exposed connection (see Figure 9.55).

Figure 9.55 Solder the Red Wire the 1N5391G Diode

- Now take the opposite end of the red wire and solder it, along with one leg of the 0.1uF capacitor, to the center post of the power connector (see Figure 9.56). Before making the solder connection, slide a piece of 3/32-inch heat-shrink tubing over the wire and capacitor lead.

Figure 9.56 Solder the Red Wire and Capacitor to the Connector's Center Post

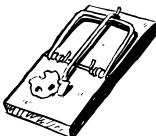


- Next, strip a short length of insulation from each end of the black wire and solder it, along with the other lead of the 0.1uF capacitor, to the outside post of the power connector (see Figure 9.57). Again, slide a piece of heat-shrink tubing over the wire and capacitor lead to help shield the connections once everything is soldered together.

Figure 9.57 Solder the Black Wire and Capacitor to the Connector's Outer Post



7. Slide the heat-shrink tubing over the terminals on the power connector, and use a heat gun to shrink them down.
8. Next, wrap electrical tape around any remaining exposed leads (such as from the capacitor) to prevent any short circuits.



WARNING: HARDWARE HARM

It's important that any exposed wire and component leads be properly shielded so that they do not come in contact with other components or wires. Because we are working on the power input circuitry of the console, electrical shorts can cause damage to the 5200 power supply or the system board, possibly rendering your 5200 inoperable.

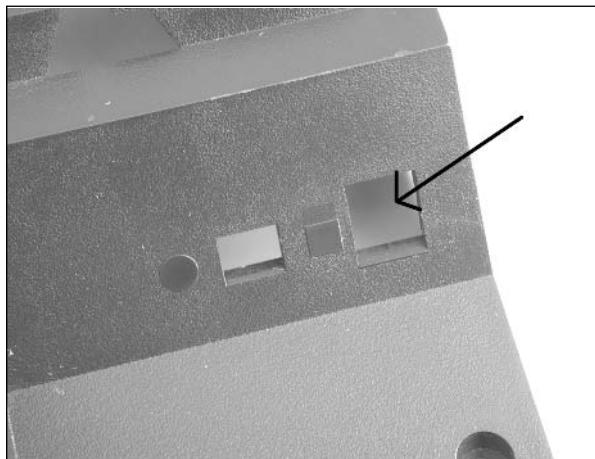
9. Cut and slide a piece of 3/16-inch heat-shrink tubing down from the opposite end of the cable over the end of the cable with the connector, covering everything from the power

Figure 9.58 Completed View of the Power Cable

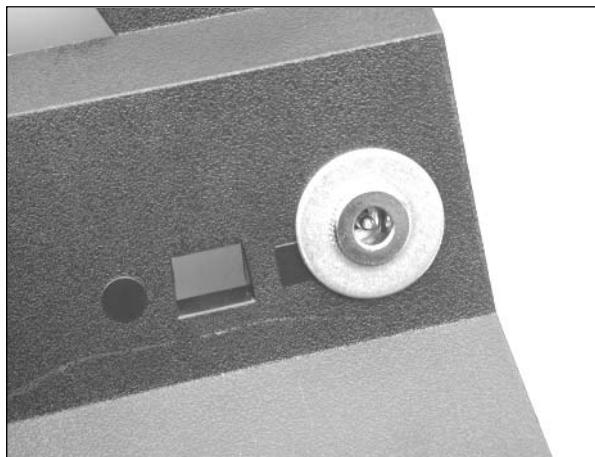


connector terminals to the capacitor. Use the heat gun to shrink this tubing. Your completed power cable should resemble the one shown in Figure 9.58.

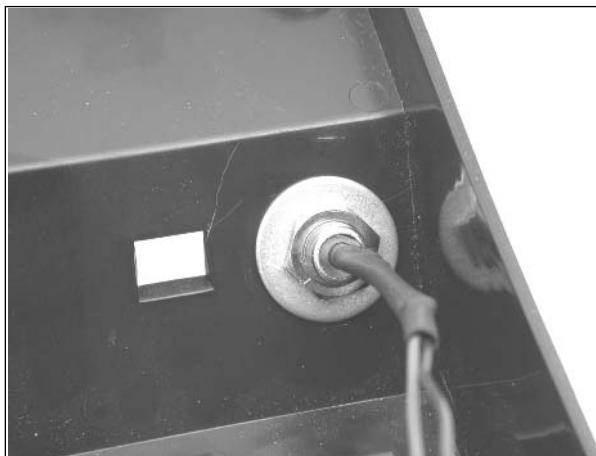
With the cable complete, we now need to attach the power connector to the bottom half of the 5200 case. We'll use the large, square hole closest to the edge of the case (see Figure 9.59).

Figure 9.59 Power Connector Hole

10. Slide one of the 7/16-inch washers over the cable and against the lip of the power connector, and feed the cable through the hole in the 5200 case from the outside (see Figure 9.60).

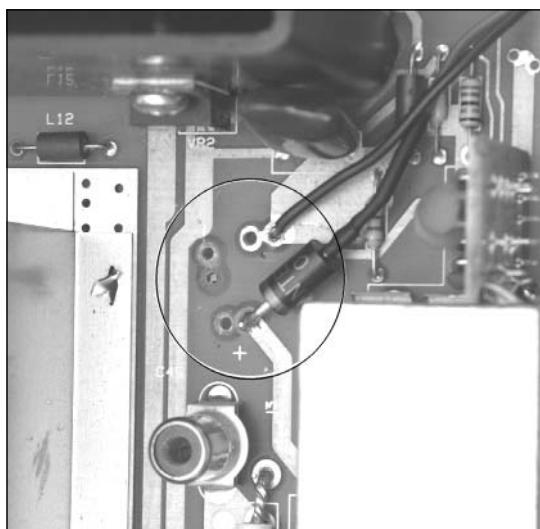
Figure 9.60 Power Connector, External View

11. Once the washer is flush with the wall of the 5200 housing, push the other 7/16-inch washer over the cable until it is resting against the interior wall of the case. Then take the nut that came with the power adapter and screw it onto the adapter's threads. Use a pair of pliers or an adjustable wrench to tighten it. The connector should now resemble the one shown in Figure 9.61.

Figure 9.61 Power Connector, Internal View

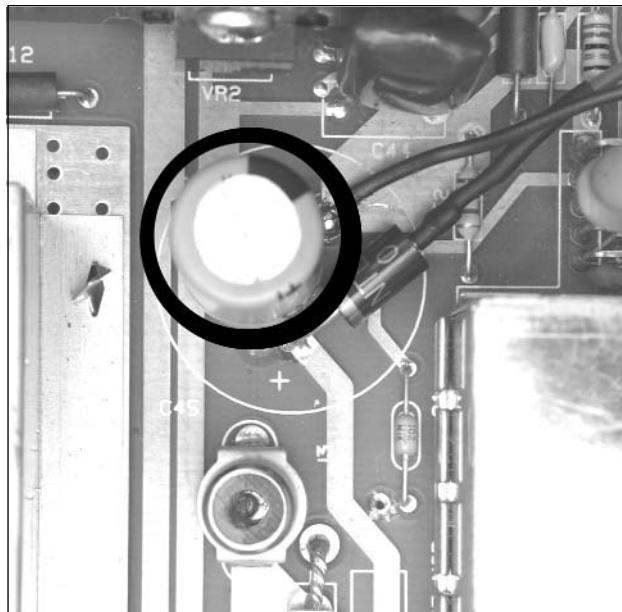
Now we can solder the other end of the newly added power cable to the Atari 5200 circuit board.

12. Solder the cathode of the 1N5391G diode (the side with the stripe) to the right solder pad of C45 (the location where you previously removed the 4700uF capacitor; it is marked with a + on the silkscreen). After securing the diode to the board, bend it down as shown in Figure 9.62, because you'll need to leave space to add the 47uF electrolytic capacitor next to it.
13. Next, solder the black wire to the right solder pad from C45 (marked with a dash [-] on the silkscreen), also shown in Figure 9.62.

Figure 9.62 Solder the Power Cable to the 5200 Circuit Board

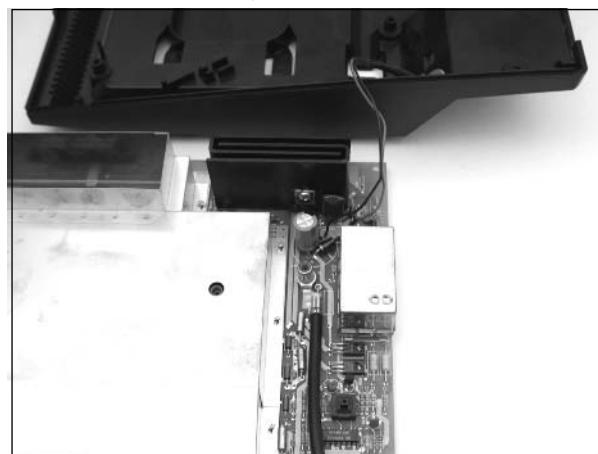
14. Now solder the 47uF electrolytic capacitor to the board using the adjacent set of solder pads for C45 (see Figure 9.63). Take care to note the polarity of the capacitor. The long lead is positive and must be placed through the solder pad marked with a “+” on the silkscreen. The shorter lead is negative (which is also denoted with a black stripe on the capacitor). Ensure that the capacitor has a good solder connection to the PCB.

Figure 9.63 Solder the Capacitor to the 5200 Circuit Board



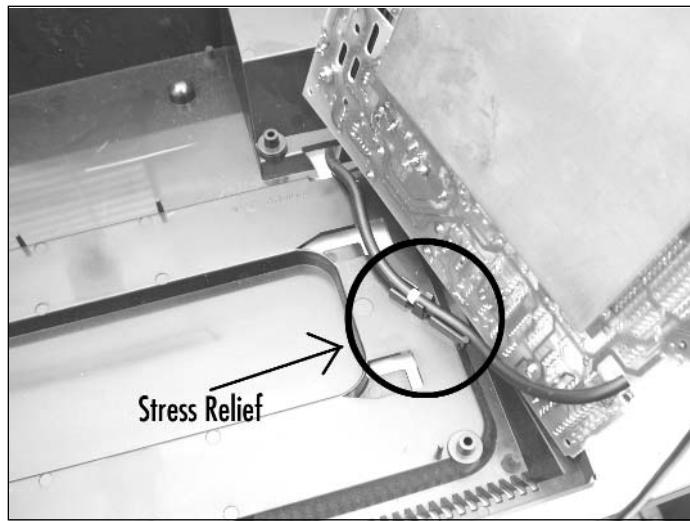
The hack is now complete, and your new circuitry should resemble Figure 9.64. You can now reassemble your 5200.

Figure 9.64 Completed Power Supply Hack, Internal View



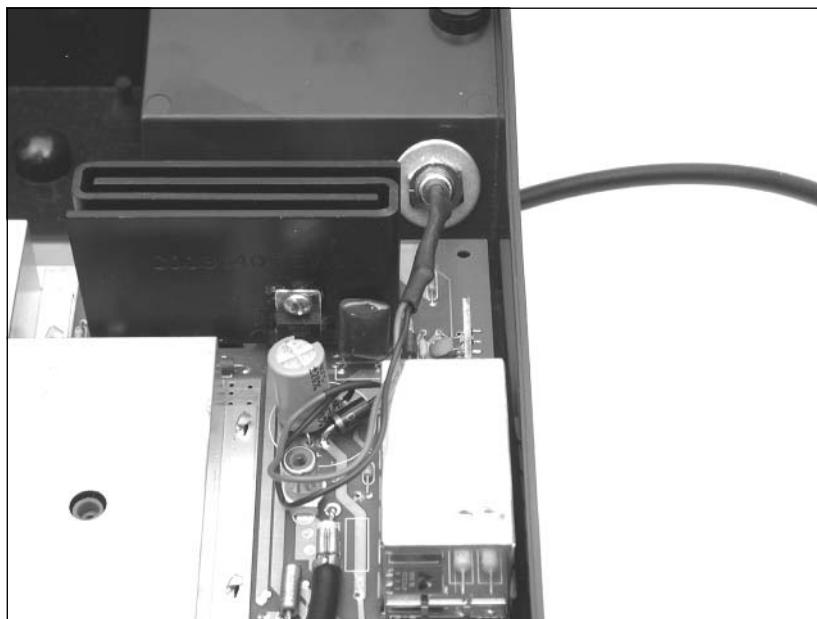
15. Start by threading the 5200 RF cable through the hole in the bottom of the case, and fasten the cable into the stress release clip (see Figure 9.65).

Figure 9.65 Attach the RF Cable to the Stress-Relief Tabs



16. With the RF cable threaded properly into the case, you can now place the 5200 PCB into the bottom half of the case (see Figure 9.66).

Figure 9.66 Modified 5200 Circuit Board Placed Back into the 5200 Case



17. Complete the reassembly of the 5200 by replacing the top cover and then attaching and tightening the seven screws.

Congratulations! You can now use a standard TV/game switchbox (or a coaxial F-type-to-female RCA adapter) with your 5200, instead of having to rely on Atari's proprietary solution. Figure 9.67 shows an external view of the 5200 case, with the power supply plugged into the newly added connector.

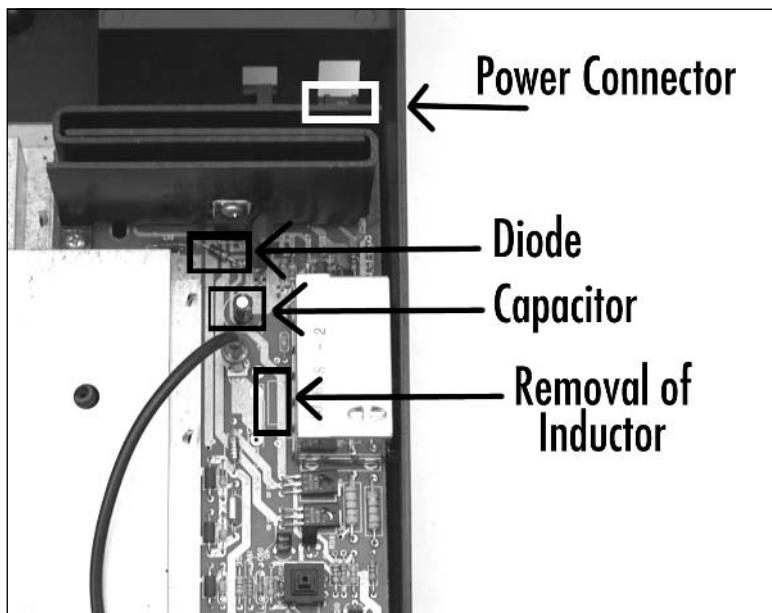
Figure 9.67 Give Me Power! Completed Power Supply Hack, External View



Under the Hood: How the Hack Works

This hack effectively mimics Atari's modification of its two-port 5200 model in which it removed the reliance on the proprietary switchbox and installed a power connector into the 5200. Figure 9.68 shows that Atari changed several power components from the four-port design: adding a power connector, adding a diode, using a smaller value capacitor, and removing the inductor.

Figure 9.68 Power Supply Circuitry of the Two-Port Atari 5200, Showing Changes from the Four-Port Version



Atari 5200 Video and Audio Upgrade Modification

Like most videogame systems released at the time, a stock Atari 5200 only supports an RF connection to the television, in which the video and audio are combined to form a single signal input to the television's antenna jack. With modern televisions, this is a less than ideal method of connecting peripherals such as videogame consoles, because significant RF interference and noise are introduced, leading to a poor-quality picture.

Thanks to an easy-to-install modification from 8bitDomain.com, you can obtain high-quality audio and video output from your Atari 5200. The 8bitDomain.com Atari 5200 Video Upgrade Board adds chrominance (more commonly called *chroma*), luminance (more commonly called *luma*), composite video, and audio jacks to the back of your Atari 5200. The chroma and luma outputs are the signals used by S-Video. Details on the common consumer-grade video signals are described in the “Atari 2600 S-Video/Audio Mod” section of the Atari 2600 chapter.

If you do this modification to a two-port Atari 5200 you will eliminate the need to use an RF switchbox at all. If you have a four-port Atari 5200, we also recommend that you perform the “Freeing Yourself from the 5200 Four-Port Switchbox” hack in this chapter so that you can eliminate your reliance on Atari’s proprietary RF switchbox. One nice feature of the Atari 5200 Video Upgrade Board is that, except for drilling holes in the plastic expansion port panel on the back of the 5200, no permanent modifications are done to the console, so you can easily remove the mod in the future.

If you don't want to purchase the 8BitDomain.com Atari 5200 Video Upgrade Board described in this hack, you can roll your own modification from scratch, as detailed in Section 4.13 of the Atari 5200 FAQ (www.atariage.com/5200/faq.html?SystemID=5200).



Preparing for the Hack

Besides a 5200 to install the video upgrade board into, you'll need to purchase the video upgrade board from 8bitDomain.com. (As this book went to press, the price of the kit was \$34.50.) The parts you'll receive include:

- A circuit board that plugs into the Atari 5200's GTIA chip socket
- A female Romex connector
- Three pre-wired, panel mount RCA video connectors
- A pre-wired, panel mount audio connector
- A probe clip that will be attached to one of the pins on the Atari 5200's POKEY audio chip
- Various nuts, screws, washers, and tie-wraps

All the parts are shown in Figures 9.69, 9.70, 9.71, and 9.72.

Figure 9.69 Contents of the Atari 5200 Video Upgrade Board: PCB, Connector, Washers, and Nuts

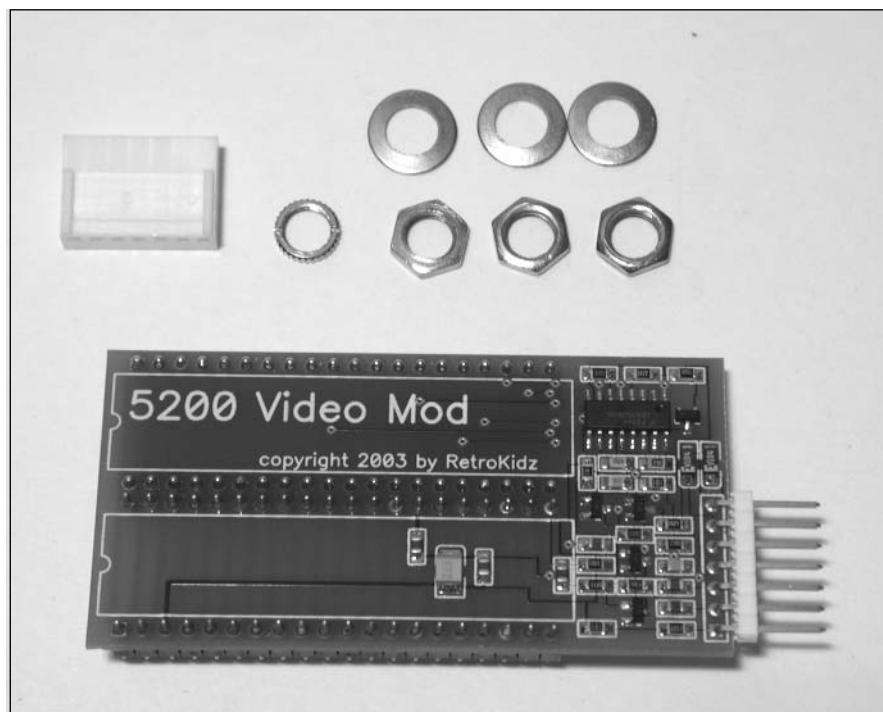


Figure 9.70 Contents of the Atari 5200 Video Upgrade Board: Pre-Wired Video Connectors



Figure 9.71 Contents of the Atari 5200 Video Upgrade Board: Pre-Wired Audio Connector

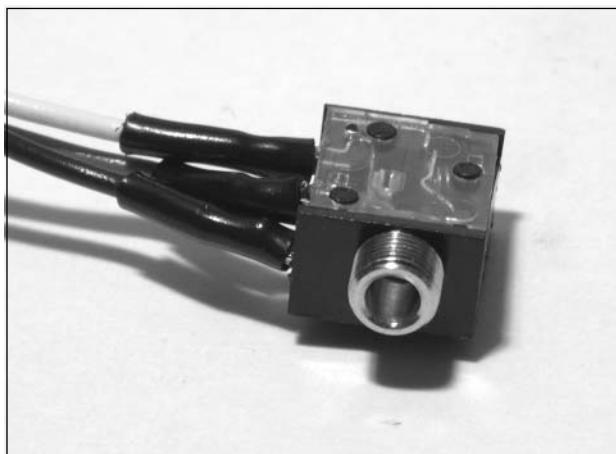


Figure 9.72 Contents of the Atari 5200 Video Upgrade Board: Probe Clip



The tools required for this hack are:

- Phillips head screwdriver, standard size
- Flathead screwdriver, standard size
- Flathead screwdriver, large size
- Needlenose pliers
- Drill and 1/4-inch drill bit
- IC extraction tool (optional)

Performing the Hack

Perform the following:

1. Remove the plastic expansion port cover on the back of the 5200 (see Figure 9.73). As you're looking at the back of the Atari 5200, the plastic cover is on the right side of the unit. Place the Atari 5200 face down on your work area and use a large, flathead screwdriver to pry the expansion cover off using the slot Atari kindly provided for this purpose. The expansion cover should snap off easily (see Figure 9.74). The removed panel cover should resemble the one in Figure 9.75.

Figure 9.73 The Atari 5200's Expansion Port



Figure 9.74 Removing the Expansion Port Cover

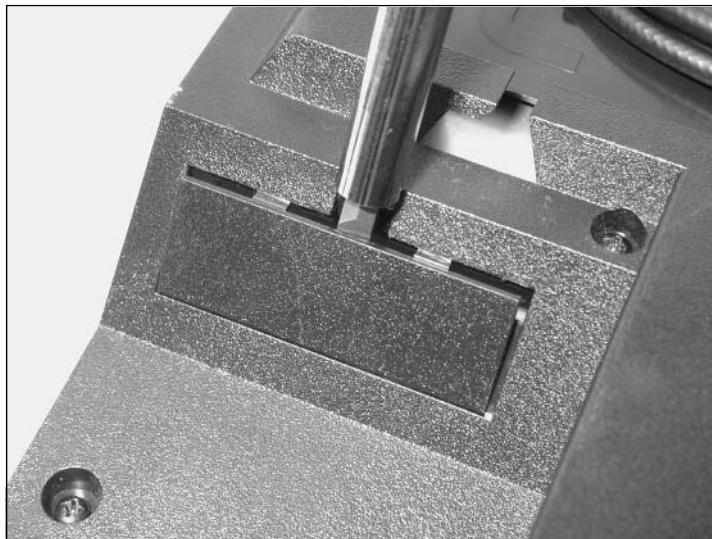


Figure 9.75 Port Cover Removed from the Atari 5200

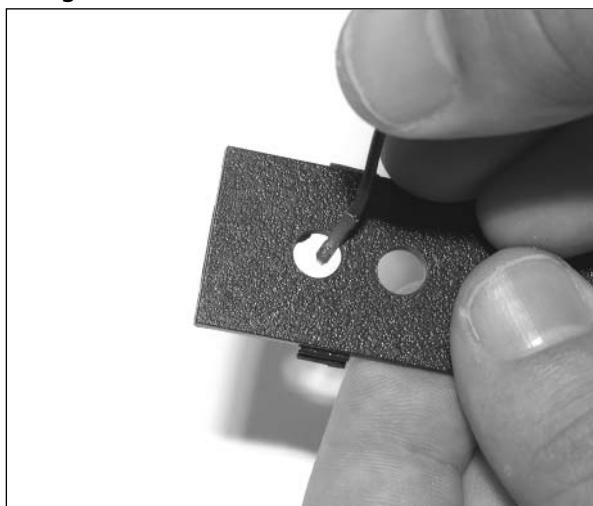


2. With the expansion port cover removed, drill four evenly spaced 1/4-inch holes into the cover (see Figure 9.76). Be sure to leave a sufficient amount of space between each hole so you won't have difficulty attaching cables to the connectors later on in the hack.

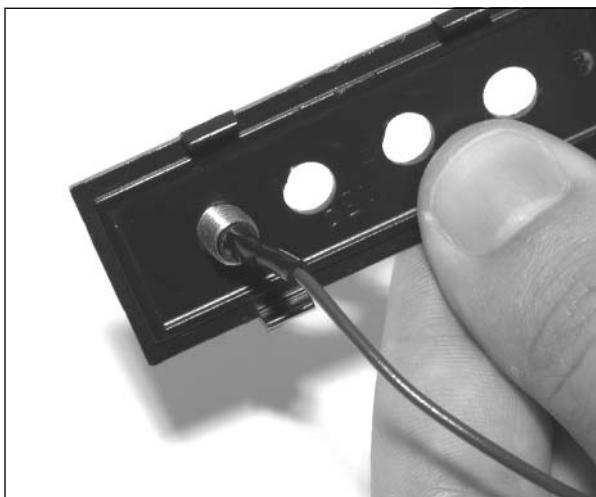
Figure 9.76 Port Cover with Holes Drilled

With the port cover prepared, we can now attach the three video connectors and single audio connector.

3. First, take the end of the blue wire opposite the RCA jack and thread it through a hole on one end of the expansion port cover (see Figure 9.77). You'll want to insert the wire through the outside face of the panel, which will result in the RCA jack facing out once you've completely pulled the wire through.

Figure 9.77 Inserting the Blue Wire into the Port Cover

4. Thread the cable completely through the hole and push the threads of the RCA jack through the hole until the jack is snug on the outside of the panel. The back of the panel (the side that will be inside the 5200 when we are done) should resemble the picture in Figure 9.78.

Figure 9.78 The RCA Connector Installed into the Port Cover

5. Soldered to the audio connector (refer back to Figure 9.71) is a black ground wire that has three metal loops attached to it. These loops must be fastened to each of the video jacks. Take the loop furthest from the audio connector and thread the blue wire through it, pushing the loop flat against the back of the plastic cover (see Figure 9.79).

Figure 9.79 Inserting the Ground Connection onto the RCA Connector

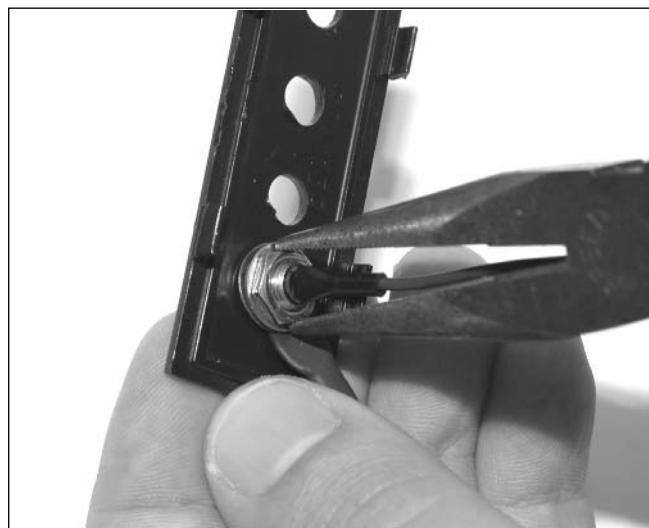
6. Next, slide a washer, followed by a nut, over the blue wire (see Figure 9.80).

Figure 9.80 Attaching the Washer and Nut to the RCA Connector



7. Use the needlenose pliers to tighten the nut until the connector is held firmly in place (see Figure 9.81).

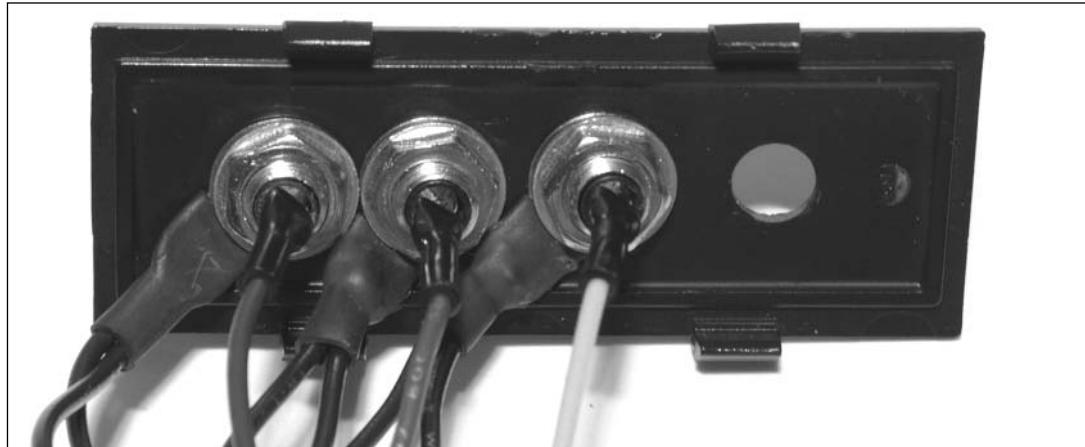
Figure 9.81 Tighten the Nut to Secure the RCA Connector



8. Repeat this process (Steps 3 through 7) for the remaining two RCA connectors. You'll want to attach the purple wire next and the yellow wire last. Try to angle the ground connections away from the bottom of the panel at a 45-degree angle, rather than having them face straight down. This will make it easier to insert the panel back into the 5200 later on in the

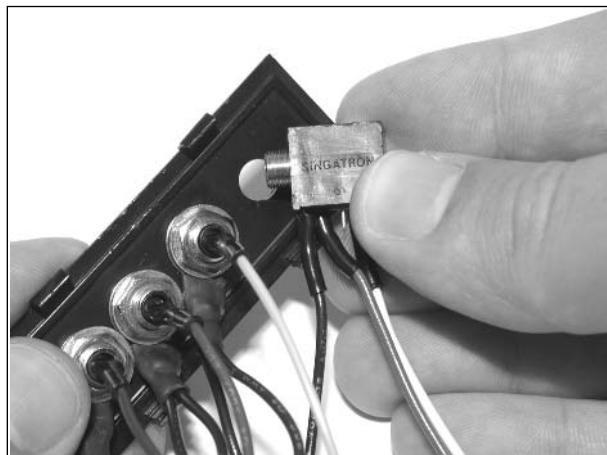
hack. After you have finished, the back of the panel should resemble the one shown in Figure 9.82.

Figure 9.82 Video Jacks Installed into the Expansion Port Cover (Left: Blue, Center: Purple, Right: Yellow)



9. Now it's time to install the audio connector into the panel cover. Place the audio jack through the back of the panel (see Figure 9.83).

Figure 9.83 Inserting the Audio Jack into the Port Cover



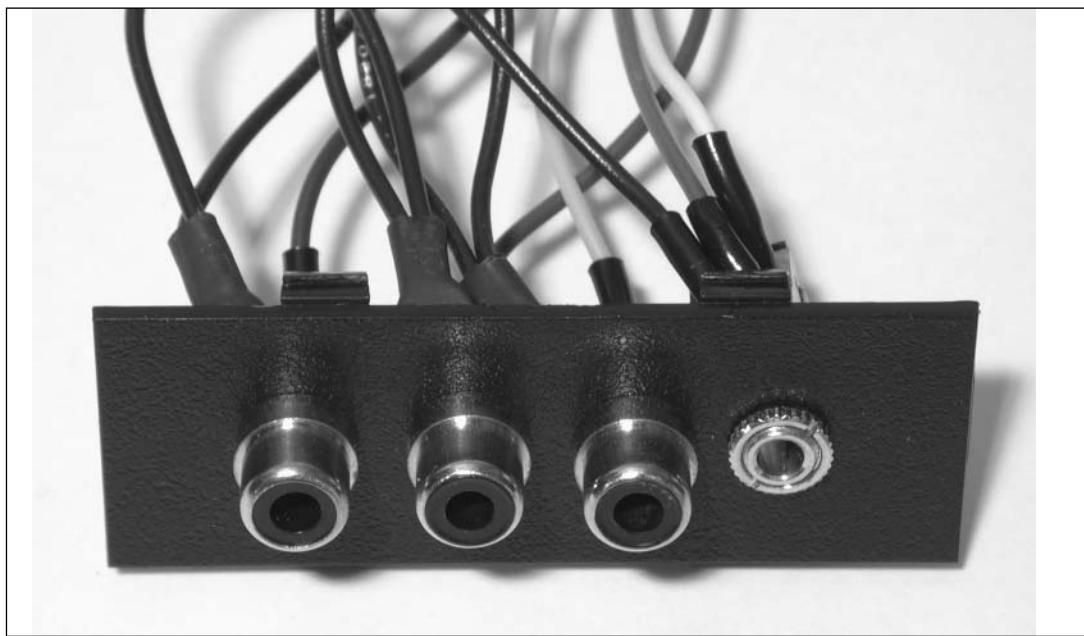
10. Use the remaining threaded nut to fasten the audio connector to the panel. This nut is attached to the audio jack from the front side of the panel (see Figure 9.84).

Figure 9.84 Attaching the Nut to the Audio Jack

11. Use a pair of needlenose pliers or a flathead screwdriver (placed into the two small grooves of the nut) to tighten the nut until the connector is held firmly into place. The attached audio jack should resemble the one shown in Figure 9.85.

Figure 9.85 Audio Jack Installed into the Port Cover

Figure 9.86 shows the inside view of the panel after all the connectors have been properly attached. Figure 9.87 shows the front view of the panel.

Figure 9.86 Assembled Expansion Port Cover, Inside View**Figure 9.87** Assembled Expansion Port Cover, Front View

Now it's time to attach the individual wires to the female Romex connector. This connector will later be plugged into the video upgrade board.

12. With the notched side of the connector away from you (and down towards your work surface), attach the wires in the order shown in Table 9.1. The seven connectors are numbered from left to right.

Table 9.1 Romex Connector Wiring

Connector	Color	Function
1	Green	POKEY Audio Clip
2	White	Audio
3	Red	Audio
4	Black	Ground
5	Yellow	Composite Video
6	Purple	Luma
7	Blue	Chroma

13. Push each wire into the Romex connector. The flat portion of the connector should be facing up, as shown in Figure 9.88. The metal connector of each wire should snap into place, and you can give the wire a gentle tug to verify that it is secured. If the metal connector does not snap into place, the wire might be inserted upside down. Remove the wire, rotate it 180 degrees, and reinsert it into the connector. The assembled connector should resemble the one shown in Figure 9.89.

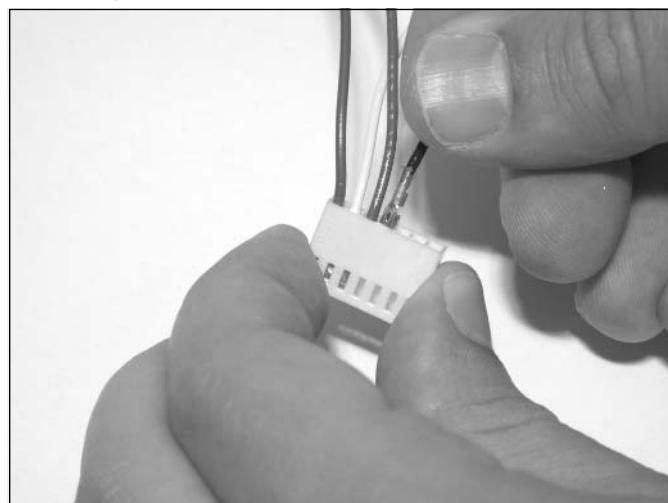
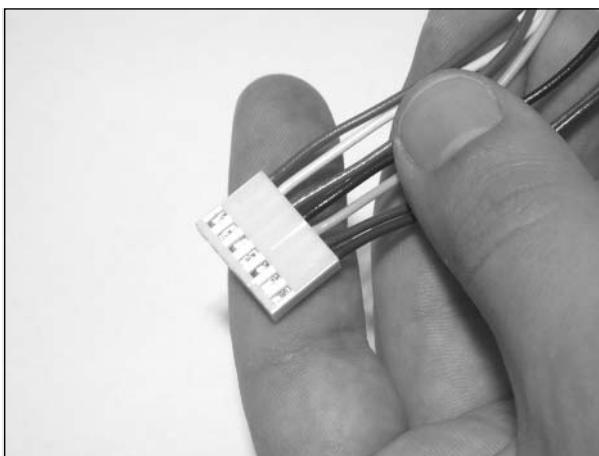
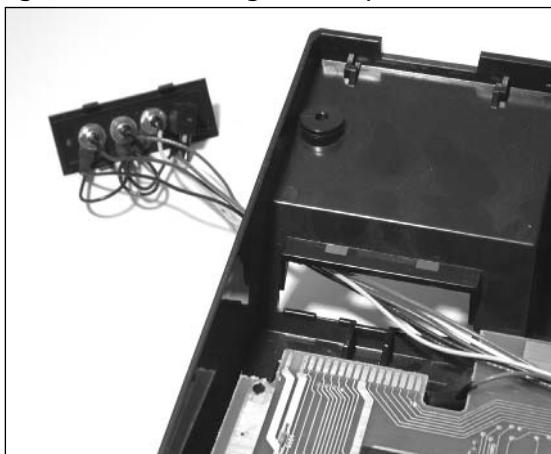
Figure 9.88 Assembling the Romex Connector

Figure 9.89 The Assembled Romex Connector (Left Wire: Green, Right Wire: Blue)

Now that we've finished our wiring, we can attach the panel cover back onto the 5200.

14. First, you need to open the four-port Atari 5200 as described in the "Opening the Atari 5200" section earlier in this chapter. This hack requires that you remove the RF shield.
15. Place the 5200 circuit board back into the bottom of the case. With this hack, you will not be reinstalling the RF shield.
16. Now feed the Romex connector through the hole in the 5200 case where you removed the plastic cover from earlier. It's important that you feed the wires through from the outside in, as shown in Figure 9.90.

Figure 9.90 Threading the Wires Through the Expansion Port Opening

17. Pull the wires completely inside the 5200 and snap the plastic expansion port cover back into place. This step is a bit tricky since the connectors are now attached to the cover. It's easiest if you snap the top part of the cover into place first (see Figure 9.91). You will likely need to

bend the ground loops up and away slightly from the panel to fit them inside the 5200. As long as nothing is in the way, the expansion port cover should snap securely into place (see Figure 9.92). Make sure that the port cover appears flush from the outside of the case.

Figure 9.91 Top of the Expansion Port Cover Clipped In

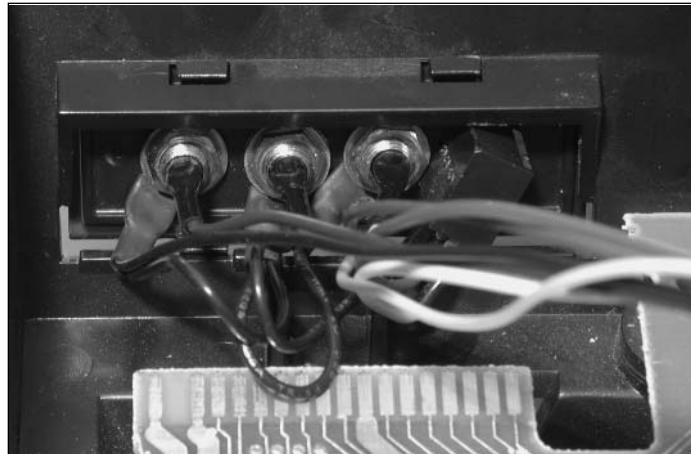
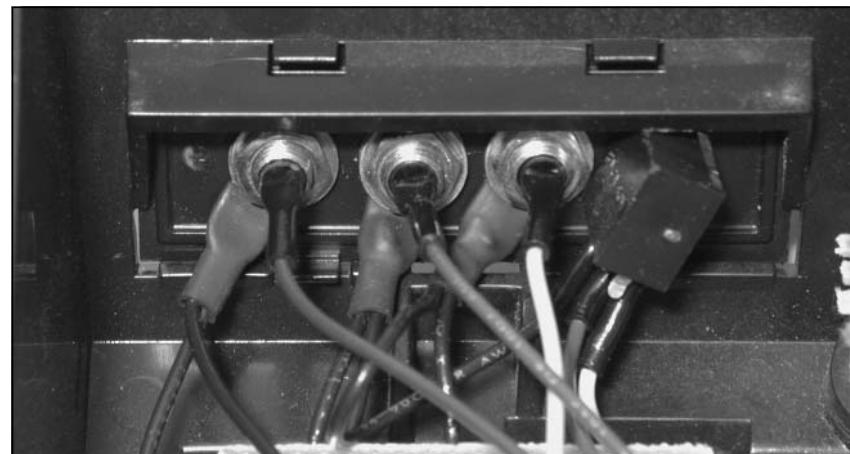


Figure 9.92 Expansion Port Cover Finally Reinstalled



With the expansion port cover complete and our Romex connector prepared, we can focus our attention on the 5200 circuit board.

18. First, we need to remove the GTIA (which stands for George's Television Interface Adapter) chip from the 5200. The GTIA generates the video output for the Atari 5200 and will be replaced later on in the hack. Figure 9.93 shows the location of the GTIA chip on the board. Using an IC extraction tool or a small flathead screwdriver, carefully remove the GTIA chip from its socket (see Figure 9.94). Be careful not to bend the pins of the chip.

Figure 9.93 Atari 5200 Circuit Board Showing the GTIA Chip

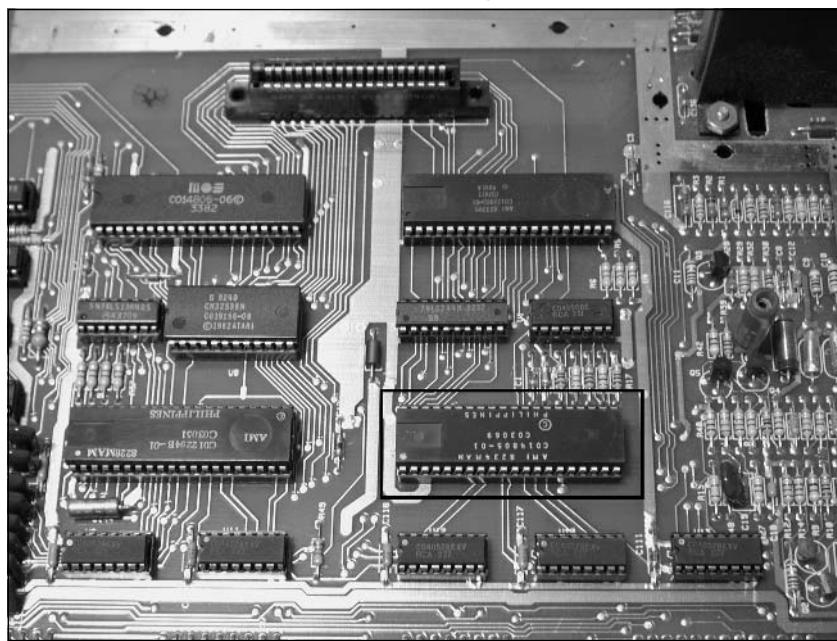
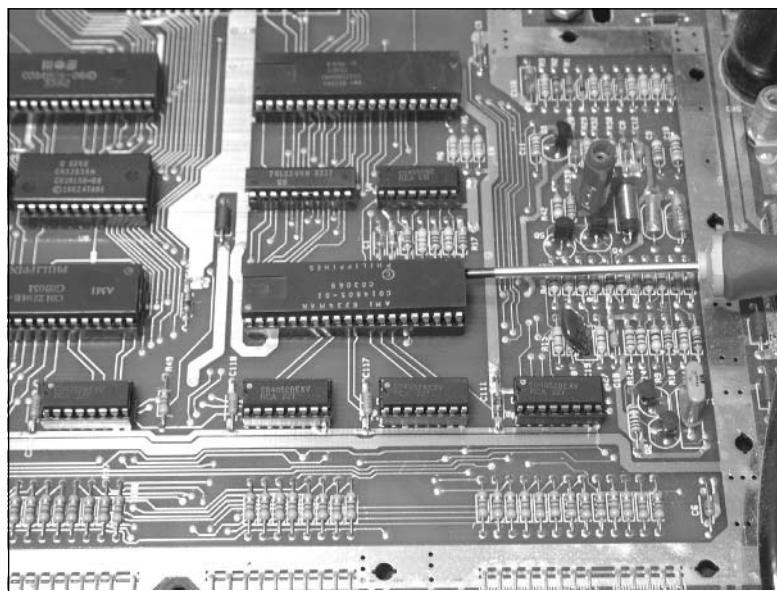


Figure 9.94 Removing the GTIA from the Atari 5200 Circuit Board



19. Turn the 5200 Video Upgrade Board over to the bottom side. You should see an empty 40-pin socket. Orient the GTIA chip so that the notch of the chip faces to the left edge of the board, as shown in Figure 9.95, and gently insert the chip into the socket. Take care that all pins are properly inserted into the holes and that no pins are bent. After you have inserted the GTIA chip, the board should resemble the one shown in Figure 9.96.

Figure 9.95 Inserting the GTIA into the 5200 Video Upgrade Board

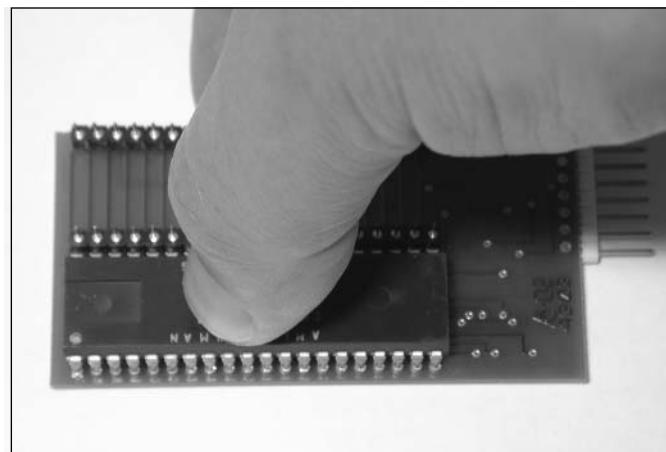
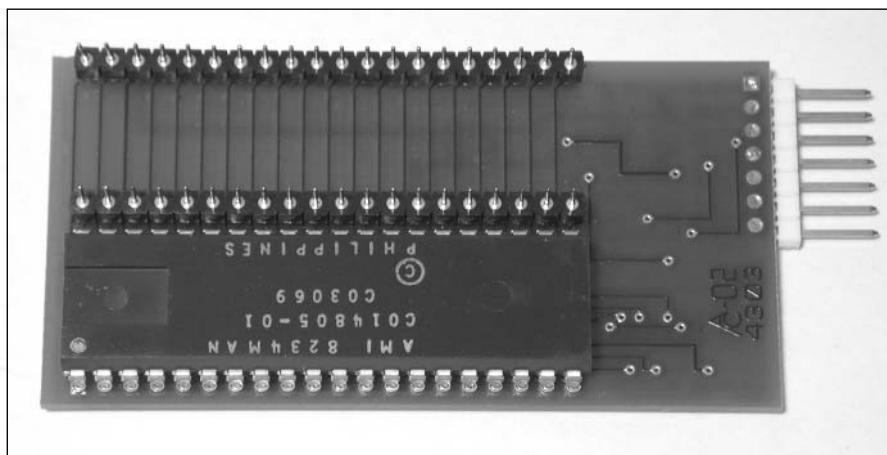
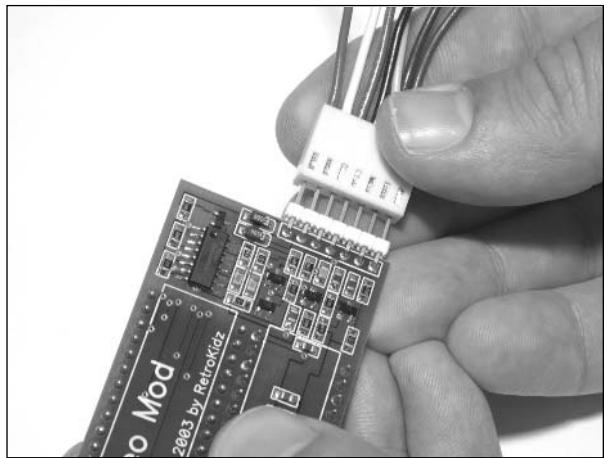


Figure 9.96 GTIA Chip Inserted into Mod



20. Holding the 5200 Video Upgrade Board right side up (so that the white text of the circuit board is facing you), slide the Romex connector onto the pins on the right side of the board (see Figure 9.97). The notch on the connector will be facing down, and the green wire should be on the top.

Figure 9.97 Attaching Romex to Board

21. Now plug the 5200 Video Upgrade Board into the socket formerly occupied by the GTIA chip. Make sure that you place the board onto the socket as shown in Figure 9.98. The notch printed on the circuit board should be aligned with the notch on the socket. The Romex connector will be on your right. Take care that all pins are properly inserted into the holes and that no pins are bent. When the 5200 Video Upgrade Board is properly installed into the Atari 5200, it should resemble the image in Figure 9.99.

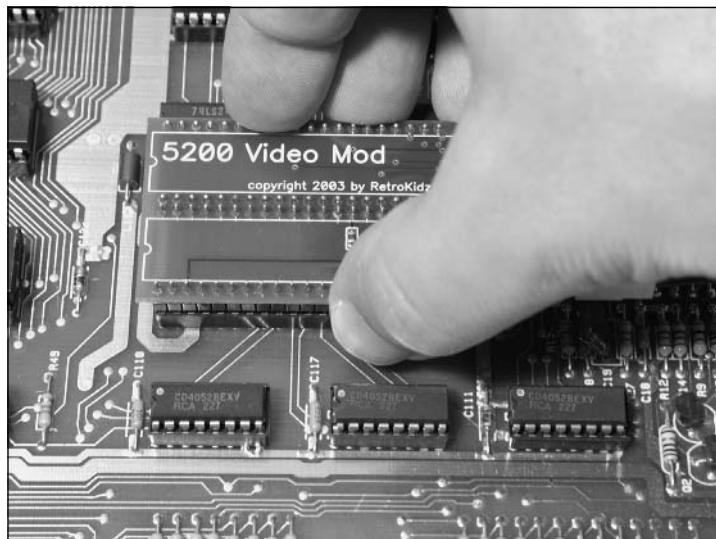
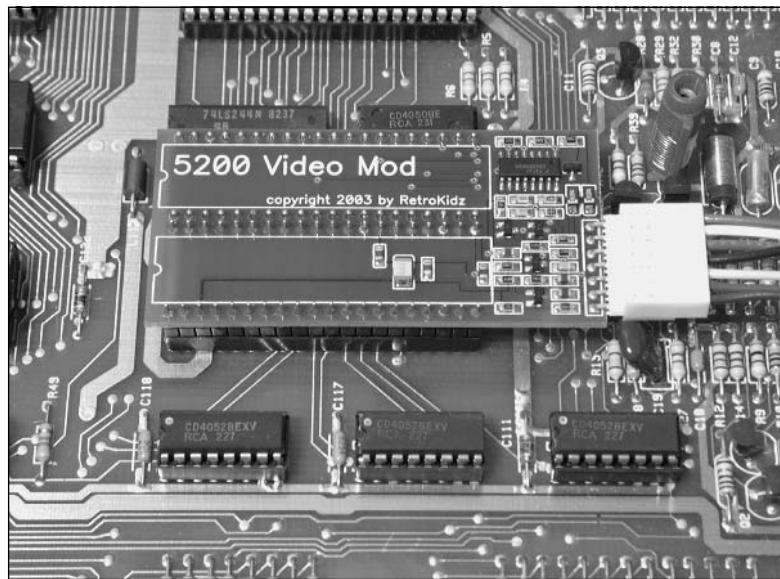
Figure 9.98 Inserting the Video Upgrade Board into the Atari 5200

Figure 9.99 The Video Upgrade Board Successfully Installed into the Atari 5200



22. Using Figure 9.100 as a guide, locate the POKEY chip on the Atari 5200 board. POKEY means “pot” and “key” referring to the paddles and keyboard, and it is responsible for generating audio in the system (among other functions). Attach the probe clip (connected to the green wire of the Romex connector) to pin 37 of the POKEY (see Figure 9.101).

Use your thumb to push on the end of the clip, as shown in Figure 9.102. This will cause a metal loop to extend from the opposite end of the clip, which you can then loop around the pin. After you have successfully snared the pin, release the pressure on the clip to retract the loop. The clip should then be securely attached to the pin. A properly attached probe clip will resemble the image in Figure 9.103.

Figure 9.100 Atari 5200 Circuit Board Showing the POKEY Chip

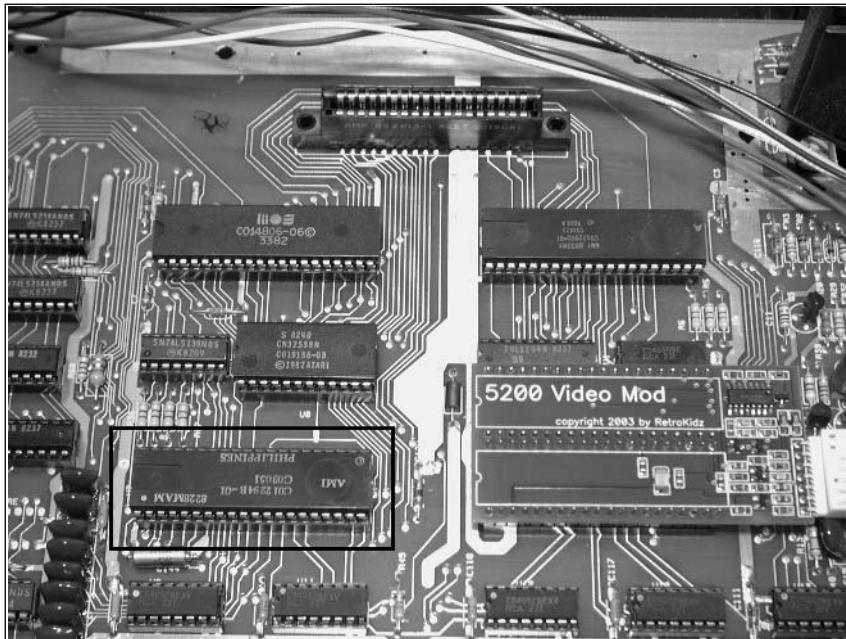


Figure 9.101 Close-Up of the POKEY Chip Showing Pin 37



Figure 9.102 Attaching the Probe Clip to the POKEY Chip

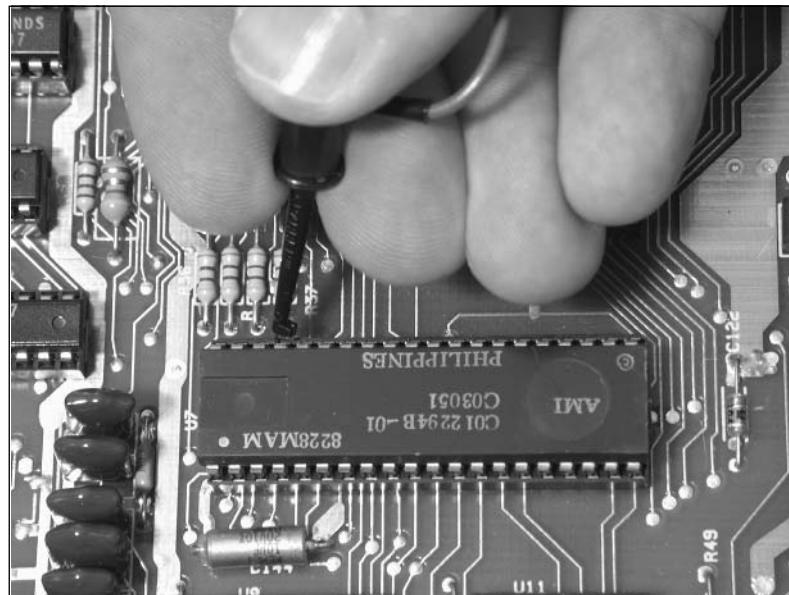
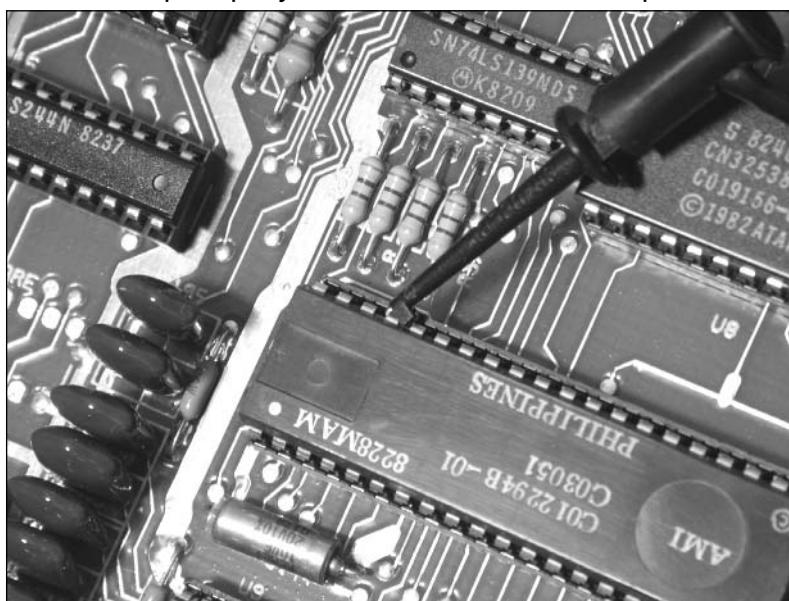


Figure 9.103 Probe Clip Properly Attached to the POKEY Chip

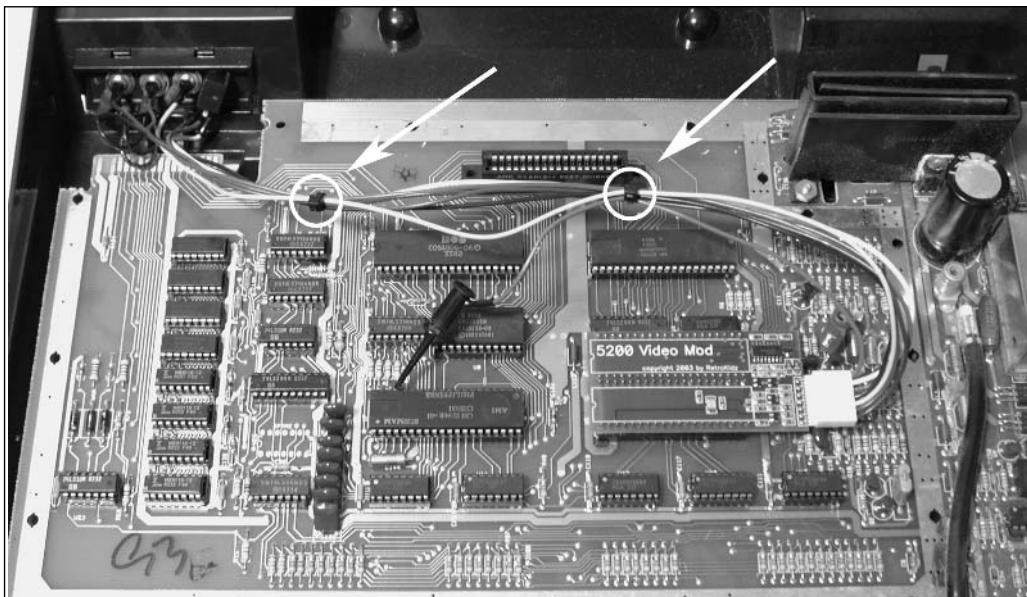


NOTE

Using the supplied probe clip on the audio output pin of the POKEY chip is something of a hack in itself. The clip could become loose or fall off if it is not put on correctly or simply due to lots of movement of the Atari 5200 system. An optional step is to use wire cutters to remove the actual clip from the wire and solder the wire directly to the top of the POKEY's pin 37. This way, you can be sure that the audio connection will remain intact. The trade-off is that you'll require a soldering iron to remove the mod if you decide you don't like it.

23. You can now use the tie wraps supplied with the 5200 Video Upgrade Board to bundle the wires together (see Figure 9.104). In particular, you want to make sure that the wires are far enough away from the cartridge slot near the top of the circuit board so that they do not interfere with insertion of a cartridge.

Figure 9.104 Bundle the Wires Together with the Tie Wraps



24. Installation of the 5200 Video Upgrade Board is now complete, and you can reassemble your Atari 5200 as described in the previous “Opening the Atari 5200” section.

You now have four new jacks on the back of your Atari 5200 (see Figure 9.105). You can use standard audio and video cables to connect your Atari 5200 to your television or monitor. You can also buy an adapter to combine the luma and chroma signals into a standard S-Video connector if your television requires it.

Figure 9.105 The New Video and Audio Connectors, Left to Right: Audio, Composite Video, Luma, and Chroma



Other Hacks

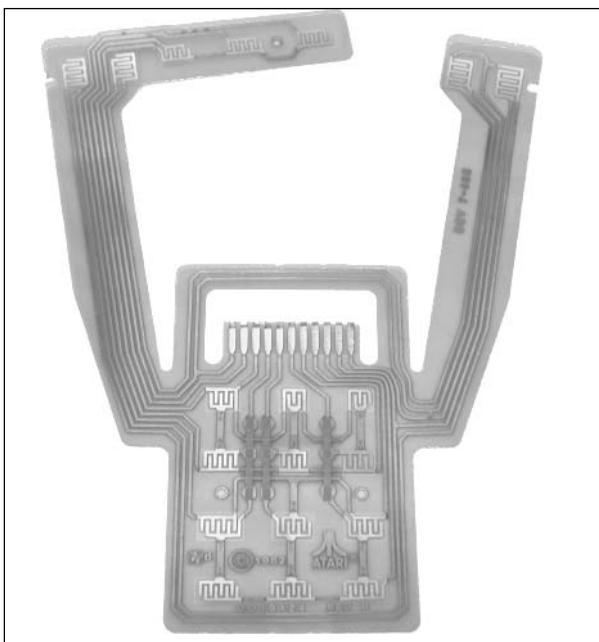
The hacks presented in this chapter are only the beginning of what you can do with your Atari 5200 SuperSystem. We've listed some of the other interesting, available hacks here; detailed instructions for many of these are available on the Internet. Whenever possible, we've pointed you toward resources for additional information.

Rebuilding Atari 5200 Controllers

As any fans of the Atari 5200 are acutely aware, the standard controllers that Atari shipped with the 5200 are lacking in several areas. The largest problem is that they are prone to failure, and it's difficult to find 5200 controllers that are 100 percent functional. Additionally, the analog joystick is noncentering, making it difficult to maintain accurate control in many games, and the fire buttons are uncomfortable and awkward to use. The most common failure is related to the buttons, which simply stop working due to poor contact inside the controller.

Fortunately, it is relatively easy to rebuild 5200 controllers using new parts from Best Electronics (www.best-electronics-ca.com). Not only does Best Electronics sell new parts to help revitalize your Atari 5200 controllers, but the company has redesigned some of these parts so that they'll last longer than stock 5200 controllers:

- **Gold Plated Rev. 9 Flex Circuit** This flexible circuit is used by the numeric keypad, the four red fire buttons, and the Reset, Pause, and Start buttons. This revision of the flex circuit uses all gold metal contacts instead of the previously used copper traces (see Figure 9.106).

Figure 9.106 Gold Plated Rev. 9 Flex Circuit

- **Gold Auxiliary Keypad, Numeric Keypad, and Fire Buttons** These three replacement buttons (see Figures 9.107, 9.108, and 9.109) contain gold-plated contacts instead of the previously used conductive rubber. The gold plating will ensure that the button makes good contact with the flex circuit and that it will not wear out as easily as rubber.

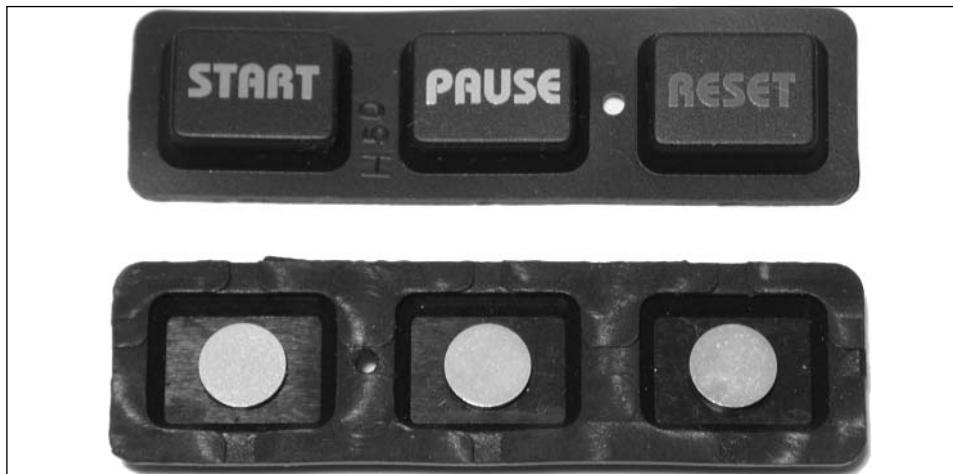
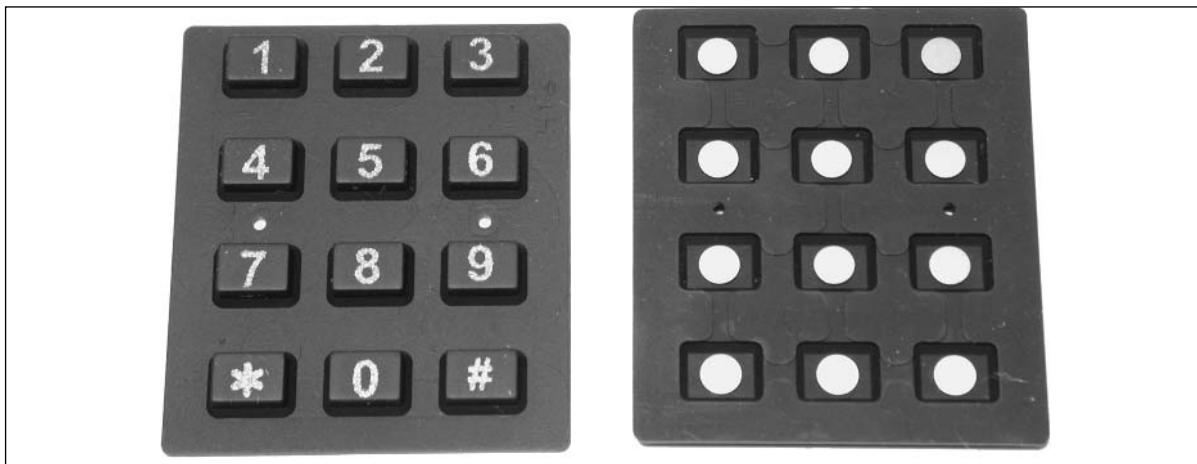
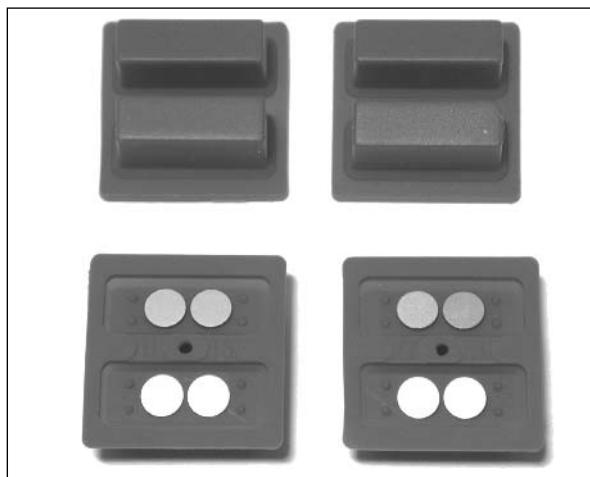
Figure 9.107 Gold-Plated Auxiliary Keypad

Figure 9.108 Gold-Plated Numeric Keypad Buttons**Figure 9.109** Gold-Plated Fire Buttons

It's always good to have working 5200 controllers, but there is an alternative that will allow you to use Atari 2600, Atari 7800, and Sega Master System/Genesis controllers on the Atari 5200. The Redemption 5200 controller adapter (see Figure 9.110) is a product designed by Pixels Past (www.pixelspast.com) that allows you to use the previously mentioned controllers with the 5200. The Redemption 5200 comes in three different models: one that works with Atari 2600/Sega controllers, another that allows use of 7800 controllers, and a third that allows the use of analog PC joysticks. They are available for purchase exclusively from AtariAge (www.atariage.com). Most games in the 5200 library do not require the use of the analog 5200 joysticks, so the Redemption 5200 is a great solution to avoid having to use the standard Atari 5200 controllers.

Figure 9.110 Redemption 5200 Controller Adapter

Atari 5200 Four-Port VCS Cartridge Adapter Fix

To make the Atari 5200 more attractive to consumers, Atari eventually developed a VCS cartridge adapter to allow Atari 2600 games to be used with the 5200. This adapter plugs into the 5200 cartridge port and contains a slot to accept 2600 cartridges, two controller ports, and the various switches found on a regular 2600 console. All two-port models of the Atari 5200 accept this adapter without problems. However, the original four-port Atari 5200 model is incompatible with the VCS adapter, except in rare cases (see the following note). Atari quickly released a technical bulletin and component kit to allow dealers to correct the problem with incompatible units.

NOTE



Not all Atari 5200 four-port models suffer from the incompatibility problem with the VCS cartridge adapter. If the serial number of your Atari 5200 contains an asterisk (*), the VCS cartridge adapter will work in your 5200 without modifications. It's unclear how many such models exist, but it seems only a small portion of four-port 5200 models fall into this category.

Homebrew Game Development

Atari 5200 homebrew development isn't nearly as popular as that for the Atari 2600 or modern consoles, but a respectable number of hobbyists are writing new games for the 5200. Because the Atari 5200 is based on Atari's 8-bit computer line, those familiar with how to program the Atari 8-bit computers can easily make the migration to the 5200.

The Atari 5200 didn't have the long life of the Atari 2600, and far fewer commercial titles were released for the system. To date, several homebrew games, such as Koffi: Yellow Kopter and Castle Crisis (see Figure 9.111), have been released for the Atari 5200, and a number of other games are in development (such as Adventure II, the unofficial sequel to Atari's classic Adventure, shown in Figure 9.112).

Figure 9.111 Atari 5200 Homebrew Games, Koffi: Yellow Kopter (Left) and Castle Crisis (Right)



Figure 9.112 Atari 5200 Homebrew Game, Adventure II Demo at PhillyClassic 5



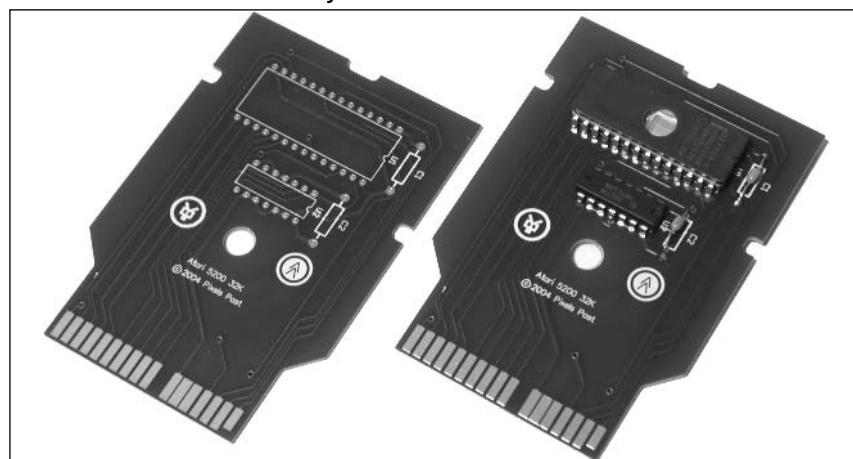
In general, games are written for the Atari 5200 in 6502 assembly language, not unlike the Atari 2600. The 5200 has much more sophisticated graphics and sound capabilities than the 2600 (courtesy of the ANTIC, GTIA, and POKEY chips), not to mention support for games up to 32KB in size without bankswitching. However, if you do not know 6502 assembly, you can get your feet wet with Atari 5200 programming by using 5200BAS, a 5200 BASIC compiler by Jeffry Johnston.

If you don't want to start programming directly for the 5200, a good way to get started is to learn how to program the Atari 8-bit family of computers. These include the Atari 400, 800, 800XL, and 130XE (to name a few). Because the Atari 8-bit computers were designed so that anyone could program them, a large number of programming books were written over the years and many magazine articles were published on how to program the 8-bit. Additionally, hundreds of type-in programs appeared in computer magazines such as *Antic* and *ANALOG*. Two fantastic resources where you can view many of these books and magazines online are at www.atariarchives.org and www.atarimagazines.com.

Although you'll ultimately want to test your creations on real Atari 5200 hardware, emulators are one of the best tools available for development purposes. Some of the excellent 5200 emulators, which run on PC, Mac, and Linux systems, include Atari800Win Plus, Virtual Super System, Jum52, and Atari800MacX. Most Atari 5200 emulators also emulate the Atari 8-bit computers. Links to each of these emulators are provided at the end of this section.

Once you are satisfied with your homebrew creation, turning it into a real cartridge is a fairly easy process. Pixels Past has created a 32K Atari 5200 PCB that accepts a standard 27256 (32K) EPROM. The only other parts you'll need are a standard 7408 AND gate and two 0.1uF capacitors. These new circuit boards fit into the original Atari 5200 plastic cartridge cases (which you can find many of online or at swap meets and yard sales). Figure 9.113 shows a bare Pixels Past Atari 5200 32K PCB on the left and a fully populated board on the right.

Figure 9.113 New Atari 5200 PCBs by Pixels Past



If you don't want to produce cartridges yourself, you can use a third party such as AtariAge to manufacture your game for you. Figure 9.114 shows the homebrew game Castle Crisis in completed form with an original label and manual design. A professionally printed box was also created for this game, but it is not pictured here.

Figure 9.114 Atari 5200 Homebrew Game: Castle Crisis



If you'd like to learn more about developing your own Atari 5200 games, we've assembled several links to get you started. When searching for information on how to program the 5200, don't forget to also look for material pertaining to the Atari 8-bit computers, since the two systems are nearly identical under the hood:

- **AtariAge, www.atariage.com** AtariAge offers many resources for the Atari 5200 developer. First, you'll want to check out the site's In Development section, which lists all the 5200 games currently in development, along with descriptions and screenshots. Aspiring programmers will want to head over to the Homebrew Development forums, which are frequented by many homebrew authors and are a great way for authors to get help and feedback while developing a game. AtariAge is the largest publisher of Atari 5200 homebrew games and frequents many classic gaming events to further promote the homebrew scene. AtariAge also sells Atari 5200 circuit boards created by Pixels Past (www.pixelpast.com) and other components for authors who want to build their own cartridges.
- **Dan B's Atari 5200 Tech Page, www.atarihq.com/danb/a5200.shtml** Dan Boris has put together a great assortment of technical information, tools, and demo source code for several Atari platforms, including the Atari 5200. Here you'll find information on the 5200 system specifications, sample source code, development software, the VSS Atari 5200 emulator, and more.

- **5200BAS Compiler, <http://lilly.csoft.net/~jeffryj/compilers/5200bas/>**
5200bas.html If you'd like to learn how to program the Atari 5200 but don't know 6502 assembly language, Jeffry Johnston has created a BASIC compiler for the 5200 that allows you to write 5200 software in BASIC and then will compile it into assembly for use with the standard TASM or DASM assemblers.
- **AtariArchives.org, www.atariarchives.org** AtariArchives.org includes the full contents of a large number of Atari 8-bit programming books, including many of the excellent Compute! books (such as *Mapping the Atari* and *Machine Language for Beginners*) and *De Re Atari* (the classic Atari 8-bit programming resource packed with technical information). You'll also find the Atari Program Exchange (APX) archive as well as the Cleveland Free-Net Atari SIG Archive here.
- **AtariMagazines.com, www.atarimagazines.com** AtariMagazines.com is a sister site of AtariArchives.org where you'll find the full text of *Antic Magazine* (all 88 issues!), a fantastic resource for those who want to learn more about Atari 8-bit computers. *Antic* featured many type-in programs, giving budding programmers the opportunity to study software submitted by other Atari hobbyists. AtariMagazines.com also features the full text of *STart Magazine* (which covered the Atari ST), as well as other magazines not directly related to Atari (such as *Hi-Res Magazine*).
- **Atari800, <http://atari800.sourceforge.net>** Atari 5200 and 8-bit emulator for UNIX, Amiga, MS-DOS, Atari TT/Falcon, SDL, and Pocket PC/Windows CE. This emulator is used as the core for many other emulators.
- **Atari800Win PLus, http://atariarea.histeria.pl/PLus/index_us.htm** Atari 5200 and 8-bit emulator for Windows.
- **Virtual Super System, <http://atarihq.com/danb/a5200.shtml#emulators>** Dan Boris's Atari 5200 emulator for MS-DOS.
- **Jum52, www.geocities.com/SiliconValley/Pines/6131/emulators/emu5200.html** Atari 5200 emulator with versions for MS-DOS and Windows.
- **Atari800MacX, <http://members.cox.net/atarimac>** Atari 5200 and 8-bit emulator for Macintosh OS X.

Atari Resources on the Web

The Atari 5200 SuperSystem has a large following of fans dedicated to sharing information and discussing the 5200. The following Web sites serve as good starting points for Atari-related resources:

- **AtariAge, www.atariage.com** Evolved from The Atari 2600 Nexus in 2001, AtariAge contains a wide variety of information about Atari game systems, featuring thousands of scans of 2600, 5200, 7800, Jaguar, and Lynx games, manuals, and boxes. In addition, you'll

find the latest Atari-related news, updates on new homebrew games in development, an online store featuring classic gaming hardware and software, and an active community discussing Atari systems in online forums.

- **Atari History Museum, www.atarimuseum.com** The Atari History Museum is a comprehensive Web site covering the entire range of Atari's history, from its early days of Pong to the days when the Atari Jaguar was state of the art. You'll find a huge wealth of historical information and pictures about Atari game consoles, computers, arcade games, and much more. The Atari History Museum got its start on the Web in 1997 and has been adding and expanding ever since. Its curator, Curt Vendel, originally ran a Bulletin Board System in the 1980s dedicated to Atari history and information.
- **AtariProtos.com, www.AtariProtos.com** Devoted to unearthing the secrets of Atari prototypes, AtariProtos.com features a growing repository of comprehensive prototype reviews for the 2600, 5200, and 7800. Each prototype is vigorously played and studied until enough is learned that a thorough review can be written. Even minor details between different versions of the game are highlighted and are often accompanied by many screenshots. AtariProtos.com is a great site for people who are curious about the oft-confusing world of prototypes.
- **B&C ComputerVisions, www.myatari.com** B&C ComputerVisions has been serving the Atari videogaming and computing communities for nearly 20 years. B&C stocks more than 5000 Atari-related products and accepts and ships orders for Atari computers, videogames, and parts worldwide. They also service most Atari products, except monitors, power supplies, and coin-operated Atari games.
- **Best Electronics, www.best-electronics-ca.com** Best Electronics, serving Atari users for 20 years, carries a large range of replacement parts and accessories for Atari game systems and computers. Featuring one of the largest inventories of Atari parts, Best Electronics has a printed catalog of more than 200 pages with over 4000 Atari products, parts, and accessories.

Atari 7800

Topics in this Chapter:

- **Introduction**
- **Blue LED Modification**
- **Game Compatibility Hack to Play Certain Atari 2600 Games**
- **Voltage Regulator Replacement**
- **Power Supply Plug Retrofit**
- **Other Hacks**
- **Homebrew Game Development**
- **Atari 7800 Resources on the Web**

Introduction

Under pressure from competitors that were eroding Atari's market share by producing competing consoles as well as clones of the 2600, Atari hastily released the Atari 5200 SuperSystem in 1982. (Hacks for the 5200 can be found in Chapter 9.) The Atari 5200 was a repackaged Atari 400 computer and did not represent development of a new game system from scratch. That would not happen until the introduction of the Atari 7800 (see Figure 10.1) five years later. Atari's original plans were to release the Atari 7800 in 1984, but those plans were shelved when the company was purchased by the Tramiel family, who wanted to concentrate on the computer side of Atari's business.

The 7800 was finally released in 1987 as a response to Nintendo's successful Nintendo Entertainment System (NES). (Hacks for the NES can be found in Chapter 7.) Ironically, Nintendo had once approached Atari about selling the NES in the United States, but Atari spurned that offer. Had the 7800 been released in 1984 as originally planned, it likely would have done quite well. But the 7800 quickly succumbed to the NES juggernaut and was unsuccessful at keeping the Atari name in the minds of videogame fans.

Figure 10.1 Atari 7800 ProSystem



Although the 7800 was released after the 5200, it actually had more in common with the Atari 2600 console than with the system it succeeded. (Hacks for the 2600 can be found in Chapter 8.) For starters, the Atari 7800 is backward compatible with 2600 games—the only system Atari would ever produce with built-in backward compatibility. The cartridges for the 2600 and 7800 are the same size, with the only discernible difference being two additional “fingers” on the cartridge connector for 7800 games. Both systems feature nine-pin controller ports and 2600-compatible controllers work just fine with the 7800. In fact, 2600 controllers can be used with 7800 games, except for those that require two independent fire buttons. The 7800 also utilizes the same hardware as the 2600 to pro-

duce sound, which unfortunately hinders the 7800 in the audio department. One game, Ballblazer, actually contains a custom Atari POKEY chip inside the cartridge, which is the same sound chip built into the Atari 8-bit computers and the 5200. Thus, Ballblazer is the best-sounding game in the entire 7800 library.

The 7800 suffered a fairly short life span, partly due to the fierce competition from Nintendo and the Sega Master System, and partly because retailers were growing wary of Atari's increasing inability to deliver on their promises. Only three games shipped at the 7800's launch, with Atari announcing many games that were never delivered. The system received poor distribution and weak support from third-party companies, with only three additional developers (Activision, Absolute Entertainment, and Froggo) producing games for the system.

Although the 7800 didn't leave much of a mark on the videogame industry, it is a popular system with Atari fans. Nearly 60 games were released for the 7800, many of them quite enjoyable to play. It's an easy system to build up a complete collection of games for, since none of the 7800 titles are exceedingly rare. And because of the system's backward compatibility with the 2600, players get the benefit of using one system to play both 7800 and 2600 games.

Hacks in This Chapter

Even though the 7800 never enjoyed the wild popularity of the Atari 2600, fans of the 7800 have scrutinized the console's internals to develop several worthwhile hacks. The first hack we'll discuss is an aesthetic mod that brings your 7800 into the modern era by installing a blue LED. We then dive into more involved hacks that can help you correct compatibility problems with the 7800, get a broken 7800 up and running again, and free yourself from Atari's proprietary 7800 power supply. The hacks are covered in the following order:

- Spice up your 7800 with a blue LED
- Apply a 2600 compatibility fix to the 7800
- Repair your 7800's broken voltage regulator
- Install a standard power jack in your 7800

If these hacks aren't enough to satiate your desire to tinker inside your 7800, at the end of this chapter we've included a list of additional hacks you can pursue. You'll also find a section on Atari 7800 homebrew development as well as a list of Web sites with 7800 content for you to explore.

Blue LED Modification

The Atari 7800 features a red LED that is illuminated when the system power is on. This hack explains how you can replace this stock LED with a blue LED to add a unique touch to your Atari 7800 system.



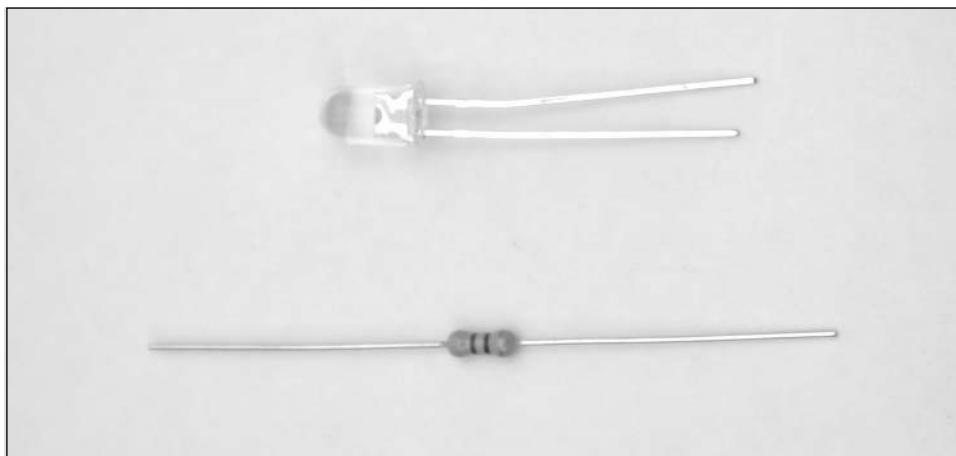
Preparing for the Hack

For this hack, you'll need two components (shown in Figure 10.2):

- A blue LED (2600mcd, 3.7V, 20mA, Radio Shack part #276-316)
- A 470 ohm, 5% resistor (Radio Shack part #271-1317)

We'll use a blue LED with a forward voltage of 3.7V and a brightness of 2600mcd at 20mA, coupled with a 470 ohm current-limiting resistor. You can experiment with different values of resistance if you'd like a brighter or dimmer LED. The resistor installed in the 7800 is 120 ohms, which, if paired with the new blue LED, results in significantly higher light output.

Figure 10.2 Required Parts



The tools required for this hack are as follows:

- A Phillips head screwdriver, standard size
- A soldering iron
- Solder sucker or solder braid
- Wire cutters
- Needlenose pliers

Performing the Hack

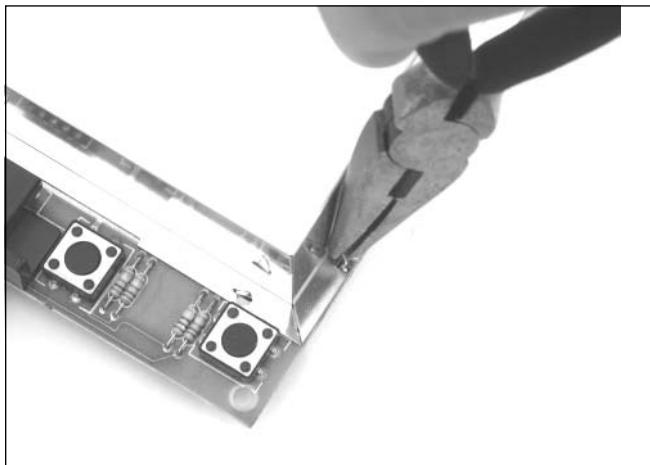
Perform the following steps:

1. First, we need to open the 7800. Flip the case upside down and remove the five screws holding the case together, as shown in Figure 10.3. The bottom-center screw might be covered by a small, round sticker, in which case you'll have to remove it first.

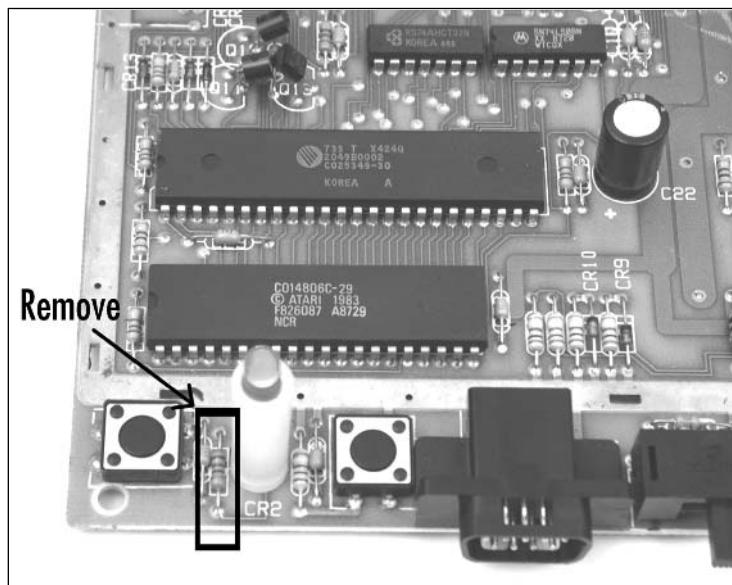
Figure 10.3 Remove the Five Screws



2. Turn the case back over, and then remove the top half of the case. The board will be sitting inside; remove it from the case.
3. Now, remove the RF shield. Use a pair of needlenose pliers to straighten all the metal tabs holding the shield in place, and simply pull the shield up off the board, as shown in Figure 10.4. Don't forget the two tabs that are between the top of the cartridge port and the metal heat sink at the top of the board.

Figure 10.4 Remove the RF Shielding

We're going to be working with the lower-left portion of the Atari 7800 circuit board, where the LED is located, as highlighted in Figure 10.5.

Figure 10.5 The Atari 7800 LED

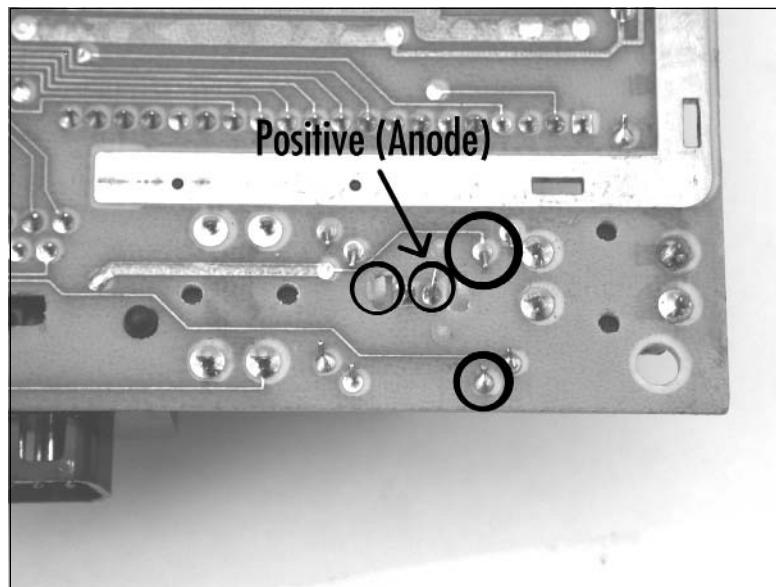
4. Next, remove the existing LED and the accompanying resistor, as highlighted in Figure 10.5. Flip the board over and unsolder the four connections shown in Figure 10.6. To remove the LED from the board, you will need to squeeze the two plastic tabs holding the plastic housing for the LED to the board.

WARNING: HARDWARE HARM

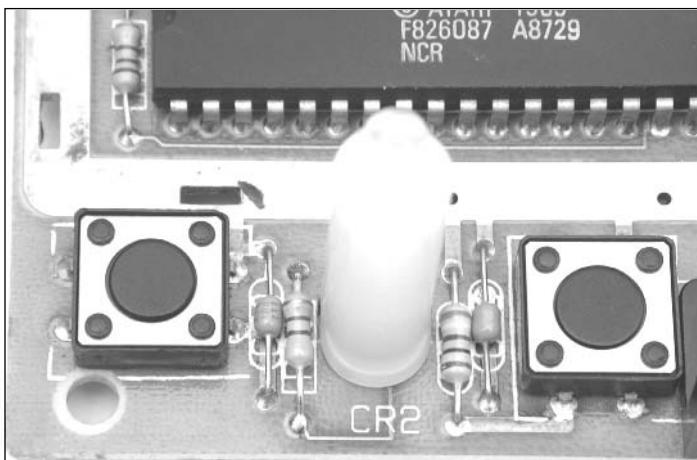
Be careful when removing the LED from the board, as you don't want to pull up the solder pads from the front side of the board. Try to remove as much solder as possible from the joints on the bottom of the board, and then carefully pull the LED up from the board while heating the connections.

- Once you have unsoldered the resistor and the LED, remove the red LED from the white plastic spacer. Place the blue LED into the spacer and then solder it to the board. The long lead of the LED is positive and must be soldered to the connection closest to the resistor you removed. The positive lead is highlighted on the back of the board in Figure 10.6.

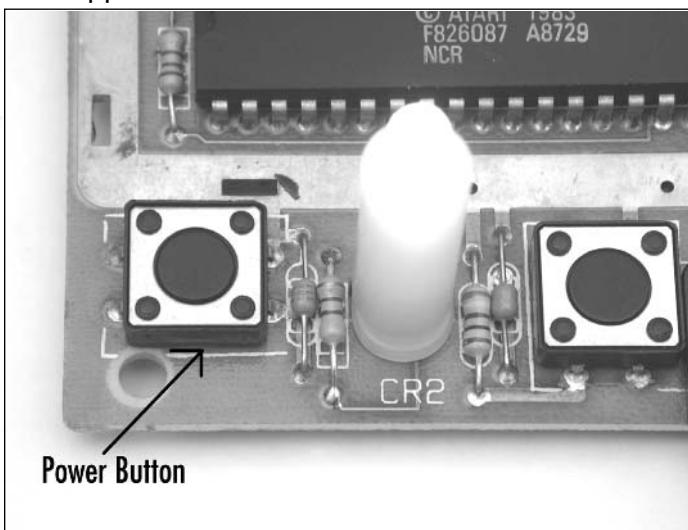
Figure 10.6 Unsolder and Remove the Resistor and LED



- Next, solder the 470 ohm resistor in the space formally occupied by the original resistor you removed. When you're done, the LED and resistor should resemble those shown in Figure 10.7.

Figure 10.7 New LED and Resistor in Place

7. If you are careful and don't have anything underneath the 7800 that could short any connections on the bottom side of the board, you can plug in the power supply and hit the power button (in the lower-left corner, as denoted in Figure 10.8) to see how the new LED looks. However, for a more accurate gauge of how the LED looks, you should view the LED inside the assembled case, since the hole for the LED on the 7800 is fairly small which reduces the light output.

Figure 10.8 Power Applied

8. You can now reassemble your 7800 by placing the board back into the bottom half of the case, fitting the top half of the case on top, and then screwing the case back together using the five screws you removed earlier (see Figure 10.9).

Enjoy the new look of your Atari 7800!

Figure 10.9 Blue LED Happiness on the Atari 7800



Under the Hood: How the Hack Works

This hack consisted of two steps: replacing a resistor and replacing the LED. If you replaced only the LED and not the resistor, the blue LED would appear much brighter than the original red LED. This is because the original current-limiting resistor is 150 ohm, allowing approximately 8.6mA to flow through the LED.

Because blue LEDs are inherently brighter than red, given the same amount of current, we need to increase the value of the current-limiting resistor to reduce the amount of current going to the LED. Typically, a red LED has a forward voltage of 2V, a green LED has a forward voltage of 2.2V, and a blue LED has a forward voltage of 2.7V to 3V—different-colored LEDs have different voltage drops, so it is not often that you can just replace one color LED with another and assume you will have the same brightness. Having the power LED too bright will draw your attention away from the TV screen when you are playing games, which can get annoying.

So how did we choose the new value of the current-limiting resistor? Well, with an application such as this (which is highly subjective), you often need to experiment to find a brightness that is appropriate. After some trial and error with various resistance values, the 470 ohm value provided a nice, bearable illumination for the power LED.

The 7800 console supplies 5 volts to the power-indicator LED. The resistor is placed in series with the LED to act as a current limiter and to let only the desired current pass to the LED. This is done to protect the LED from exceeding its maximum current specification (in this case, 20mA). By adjusting the value of the resistor, we adjust the amount of current going to the LED, which in turn changes the brightness of LED illumination.

Armed with the specifications of the blue LED (in our case, a forward current of 20mA and a forward voltage of 3.7V) and our new resistor value, we can calculate the amount of current flowing through the LED by using Ohm's Law, which states:

$$\text{Voltage} = \text{Current} * \text{Resistance}, \text{ or } V = I * R$$

To calculate current, we use the formula as follows:

$$\text{Current} = \text{Voltage} / \text{Resistance}, \text{ or } I = V / R$$

Plugging in the values for the known voltage and resistance, we get:

$$I = (5V - 3.7V) / 470 \text{ ohms} = 0.0027A = 2.7mA$$

Using this formula, we can see that we are driving the LED at 2.7mA, which is only a fraction of the maximum allowable current of the LED, significantly reducing its brightness. Feel free to experiment with different values of resistance to increase or decrease the light output of the LED.

Game Compatibility Hack to Play Certain Atari 2600 Games

The Atari 7800, Atari's third-generation cartridge-based game console, will play Atari 2600 games without an additional adapter (unlike Atari's second-generation system, the 5200, which required the VCS Cartridge Adapter to play 2600 games). However, some cartridges will not work on many 7800 consoles, including Activision games (Space Shuttle, Robot Tank, and Decathlon) that use the FE bankswitching method, as well as the Supercharger and Cuttle Cart, two RAM-based devices that allow games to be loaded into their memory.

Atari added a circuit to later versions of the Atari 7800 to fix a compatibility problem with the 2600 game Dark Chambers but in the process broke compatibility with other titles, as previously mentioned. This hack will show you how to modify your 7800 to fix the compatibility problems and let you play more 2600 games. However, doing so will prevent the 2600 version of Dark Chambers from playing (but the 7800 version will still work fine).

NEED TO KNOW ... WHAT IS BANKSWITCHING?



When Atari designed the 2600, they gave it a 12-bit wide external address space and an 8-bit wide data bus, limiting 2600 games to a maximum size of 4KB without any special circuitry. At the time the 2600 was designed, Atari's hardware engineers felt that 4K would be more than enough for any 2600 game.

To get around the cartridge size limitation, Atari introduced *bankswitching* with the release of the game Asteroids. Bankswitching is a method used to overcome a memory size limitation (such as 4KB for the Atari 2600) and allows a larger-size memory device to be used with a cartridge, thus providing more data storage for the system. Bankswitching requires specially designed logic circuitry on the cartridge to handle the specific schemes. In the case of Asteroids, Atari used what is now known as an F8 bankswitching scheme (for 8K), although the company would later create games that used F6 (for 16K games) and F4 (for

32K games). Third-party companies such as Activision would develop their own schemes independently, leading to a wide variety of bankswitching methods, including the FE method employed by Activision that causes problems on many Atari 7800 consoles.

Bankswitching generally works by defining “hot spots” used by the game program to switch between memory sections (or banks) of the game ROM. Atari’s F8 method is used to switch between two 4K banks of memory. Having 2 4KB banks gives the system a possible 8KB of accessible data space. If the game accesses address 1FF8, the first 4KB bank is used. If the game accesses address 1FF9, the second 4KB bank is used. This is the simplest bankswitching method and the most commonly implemented.

For a thorough explanation of the majority of known bankswitching methods, refer to www.tripoint.org/kevtris/files/sizes.txt, written by Kevin Horton.

Preparing for the Hack



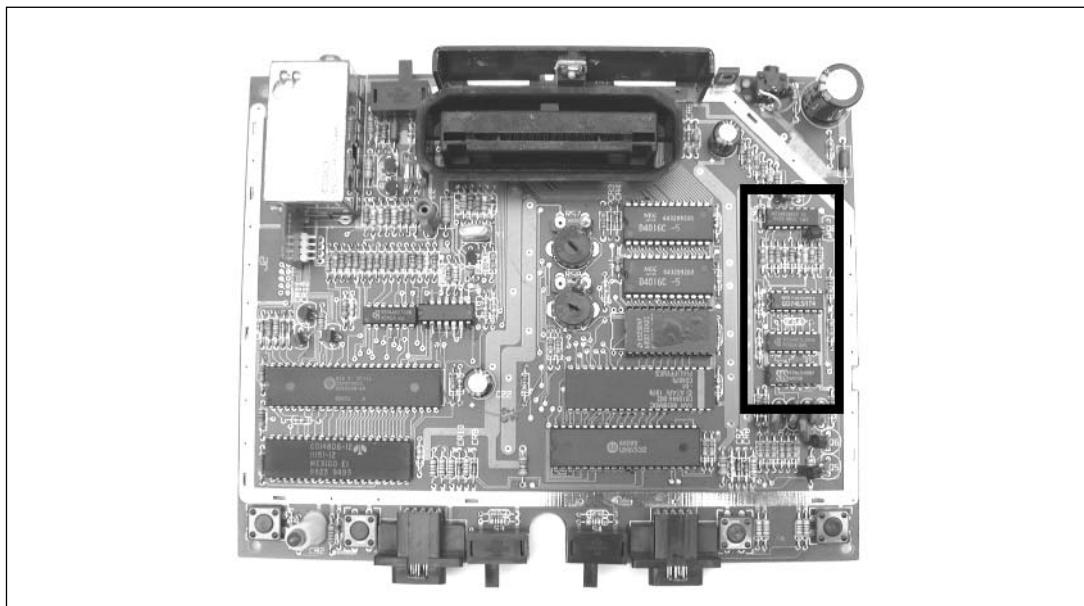
This is a quick and easy hack. The required tools are as follows:

- A Phillips head screwdriver, standard size
- Wire cutters and stripper
- Needlenose pliers
- A soldering iron (optional, if you want to remove the component instead of just clipping it out)
- Solder sucker or solder braid (optional, if you want to remove the component instead of just clipping it out)

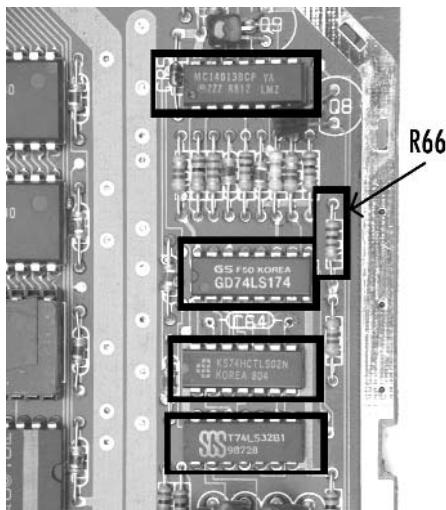
Performing the Hack

Perform the following:

1. Open the 7800 and remove the RF shield, as described in the blue LED modification hack earlier in this chapter. This will give you access to the 7800 circuit board.
2. Once you have the RF shield removed, you need to verify that this compatibility fix applies to your particular 7800. The area we are interested in is depicted in Figure 10.10.

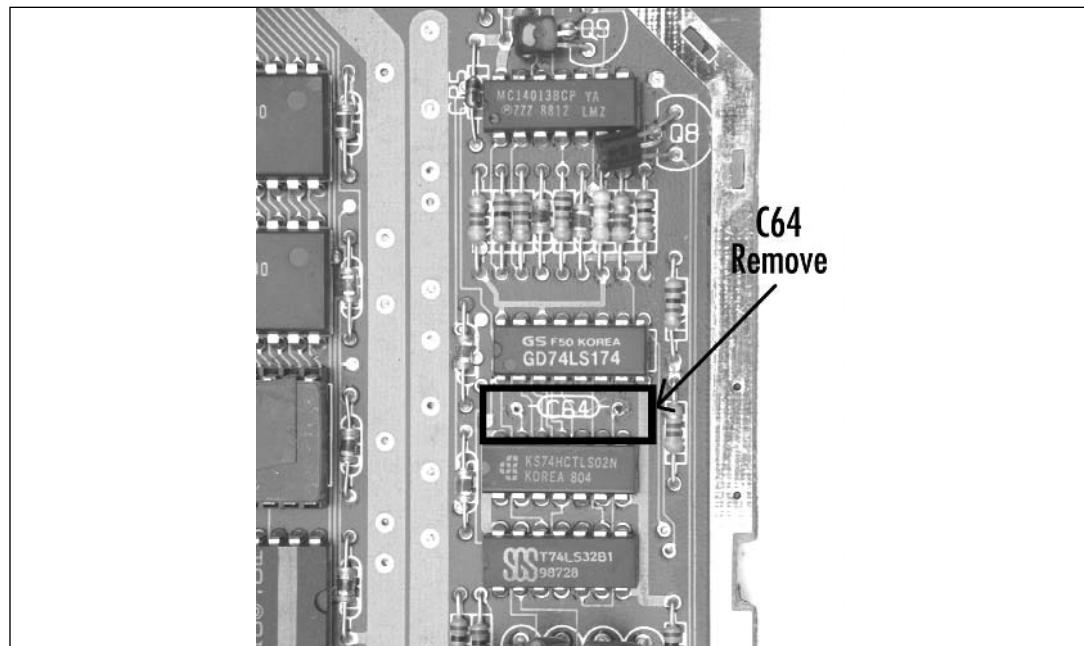
Figure 10.10 Atari 7800 Board Showing the Area of Interest

3. There are two things to look for to make sure you can perform the 7800 compatibility hack on your system. First, verify that there are four ICs in a row along the right side of the board (see Figure 10.11). If only three ICs are present, this mod does not apply. Second, verify that R66 is installed on the board. R66 is the rightmost resistor above the 74LS174 chip. There should be seven resistors and two capacitors lined up in this row. If there are not, this modification does not apply to your 7800.

Figure 10.11 Verify That Your 7800 Has These Components

4. If your board qualifies, all you need to do is clip the capacitor at location C64 from the board, as highlighted in Figure 10.12. You can also unsolder the capacitor instead of clipping it out, if you choose. If you're feeling ambitious, you can install a switch to enable and disable this capacitor for on-demand compatibility with the 2600 version of Dark Chambers.

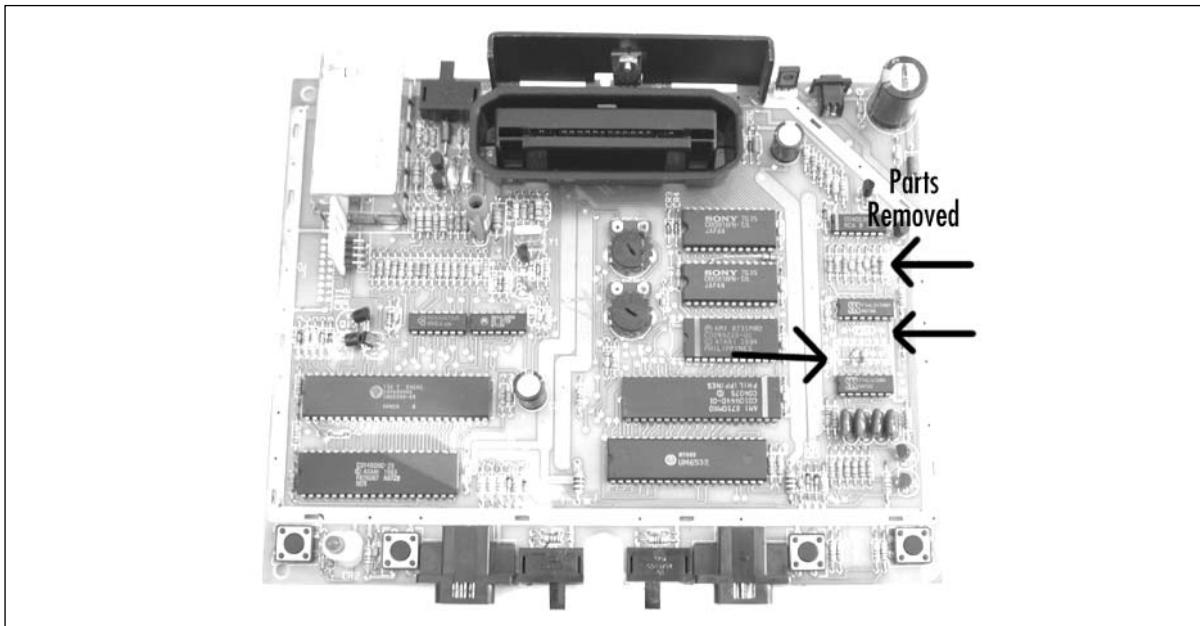
Figure 10.12 Remove the Capacitor at Location C64



5. Now you can reassemble your 7800. First replace the RF shield, making sure you retwist the metal tabs that keep the two halves of the RF shield together. Then place the board in the bottom half of the 7800 shell, attach the top half of the case, and screw the case back together with the five screws you removed earlier. You can now test some of the games known to have problems with the Atari 7800, including Activision's Decathlon, Robot Tank, and Space Shuttle. In addition, the Supercharger and Cuttle Cart cartridges should also work without problems.

Under the Hood: How the Hack Works

This hack works by disabling a timing circuit that Atari installed in order to fix a compatibility issue with Atari's 2600 version of Dark Chambers. This timing circuit involves the 74LS02 (a Quad 2-Input NOR gate), the logic chip directly below the C64 capacitor you clipped or removed. Some Atari 7800 systems are missing this chip, as well as C64 and R66. Figure 10.13 shows a 7800 circuit board that never had these parts installed.

Figure 10.13 7800 Board Without the Dark Chambers Circuit

Voltage Regulator Replacement

A common problem for the Atari 7800 is for the 5V voltage regulator to fail. This will render your 7800 inoperable, since the system will not receive power. It's even possible for the regulator to completely snap off the board due to the weight of the attached heat sink. On many 7800s, the heat sink is not soldered into place, allowing it to wobble back and forth, which causes stress and wear on the pins.

This simple hack involves opening the 7800, removing the existing voltage regulator, and soldering a new one into place.



Preparing for the Hack

If your 7800 is not powering up, it could be for several different reasons, one of which could be a voltage regulator failure. If your voltage regulator seems to be functioning properly, there is no need to replace it—you'll need to troubleshoot your system further to diagnose the problem.

If you do need to replace the voltage regulator, you'll need a replacement 7805 5V, 1A linear voltage regulator (Radio Shack part #276-1770), as depicted in Figure 10.14. This is a common part that you should be able to find at most electronics stores. You'll also want some heat-sink grease (Radio Shack part #276-1372A) to place between the heat sink and the regulator; this grease assists in proper heat transfer from the regulator to the heat sink.

The required tools for the hack are as follows:

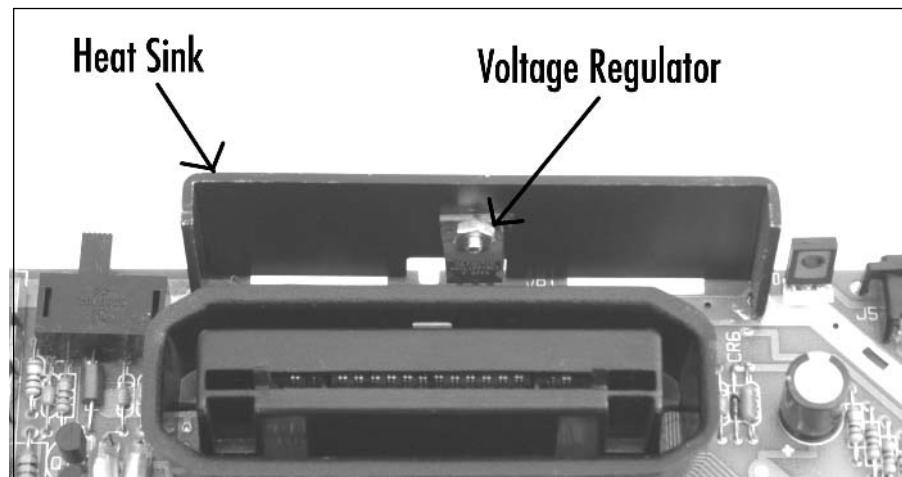
- A multimeter
- A Phillips head screwdriver, standard size
- Wire cutters and stripper
- Needlenose pliers
- A soldering iron
- Solder sucker or solder braid

Performing the Hack

Perform the following:

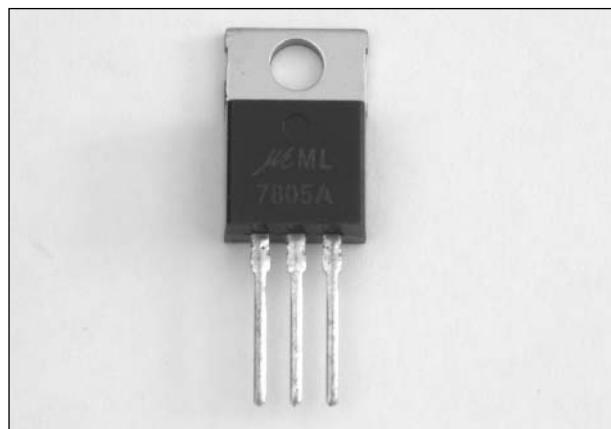
1. First open the 7800 and remove the RF shield as described in the blue LED modification hack earlier in this chapter. This will give you access to the 7800 circuit board. The voltage regulator is located at the top of the board, immediately above the cartridge port, as shown in Figure 10.15.

Figure 10.15 Atari 7800 Voltage Regulator and Heat Sink



2. You'll first want to test the voltage regulator to see if power is even reaching it, and then check to see what the output voltage is. To do this, plug the 7800 power supply into the board, and then apply power to the system by pressing the power button located in the lower-left corner of the board.

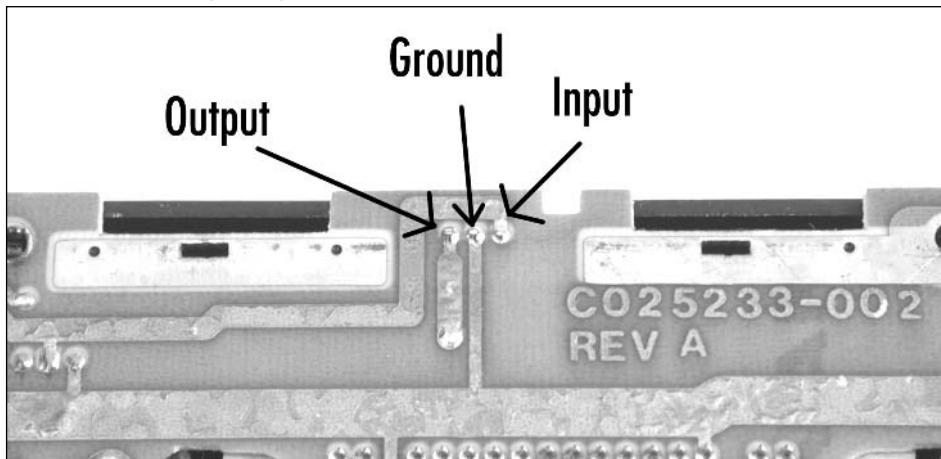
Figure 10.14 7805 Voltage Regulator



Then, carefully flip the board over and use a multimeter to measure the voltage on the input and output pins of the regulator. Figure 10.16 highlights the relevant pins on the reverse side of the 7800 board. You should measure approximately 12V on the input side and 5V on the output.

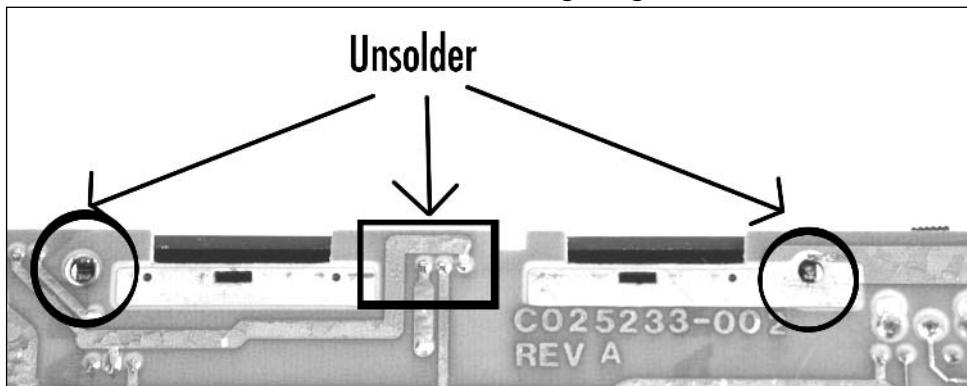
If you do not measure a voltage on the input to the regulator, the regulator might be loose or broken on the board (or your problem lies elsewhere). If you measure a voltage on the input side but do not measure a voltage on the output side (or if your output voltage is not hovering around 5V), this hack is for you!

Figure 10.16 Voltage Regulator Connections



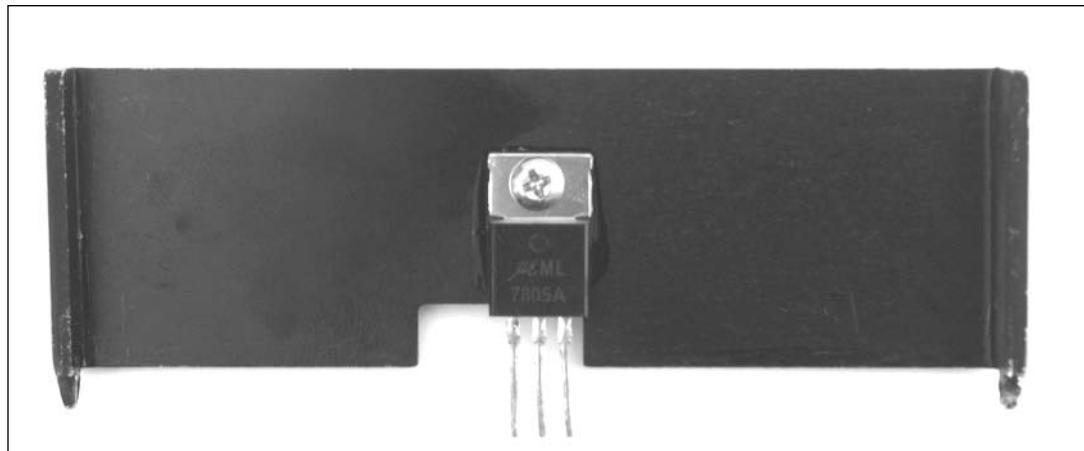
- To continue with the hack, turn the 7800's power off and unplug the power adapter from the board. To remove the voltage regulator, unsolder the three connections for the voltage regulator on the backside of the board. If the heat sink is also soldered in place, unsolder that from the board as well. Figure 10.17 shows which connections you want to unsolder.

Figure 10.17 Unsolder the Heat Sink and Voltage Regulator

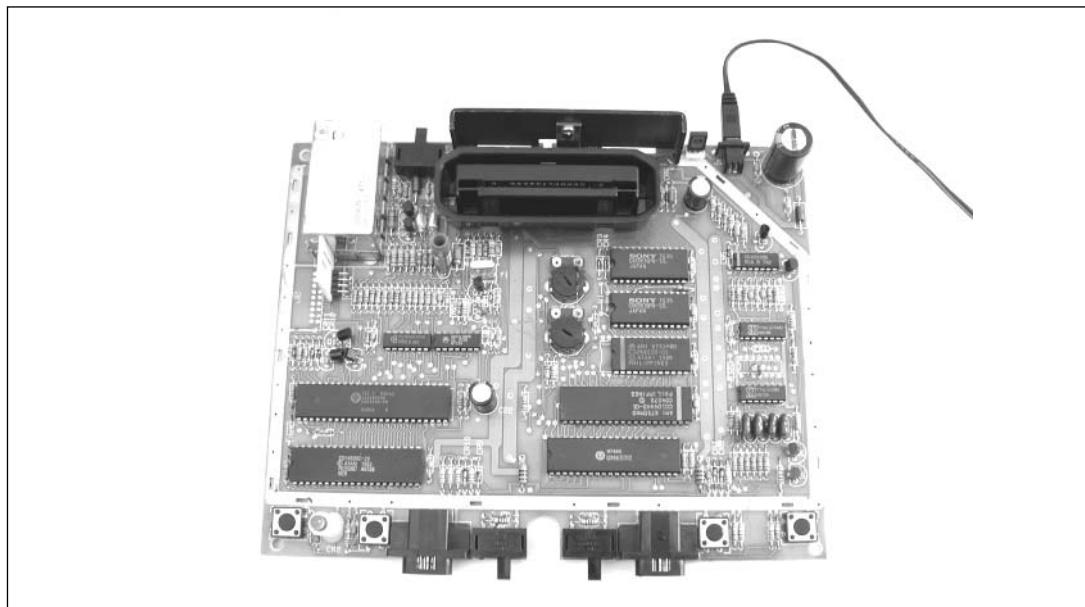


4. Flip the board back over so the component side is visible, and lift the voltage regulator and heat sink as one piece from the board. Unscrew the voltage regulator from the heat sink. Discard the regulator, but keep the screw, nut, and washer. Apply a small amount of the heat sink compound to the backside of the new voltage regulator. Attach the new regulator to the heat sink using the screw, nut, and washer you saved, as shown in Figure 10.18. The washer and nut should be used on the side of the heat sink opposite the voltage regulator.

Figure 10.18 New Voltage Regulator Attached to Heat Sink



5. Carefully place the leads for the voltage regulator through the three holes in the circuit board, as well as the mounts for the heat sink through the larger holes intended for them. Flip the board over and solder the voltage regulator in place. You should also solder the heat sink in place so that it is more secure, but you will need to use a higher-temperature soldering iron, since the heat sink will quickly draw away heat as you attempt to solder it. Trim the leads of the voltage regulator.
6. You can do a quick test to see if your repair worked by plugging the power supply into the board and then turning the power on (see Figure 10.19). Before you do this, make sure that there are no loose screws, wires, or other conductive materials underneath the circuit board. If the only problem with your 7800 board was the voltage regulator, it should turn right on!

Figure 10.19 Testing the Repair

- Reassemble the 7800, first by replacing the RF shield, placing the board in the bottom half of the 7800 shell, attaching the top half, and then screwing the case back together with the five screws that you removed at the beginning of the hack.

Under the Hood: How the Hack Works

If you've opened many electronic devices (such as the Atari 7800 described in this hack), you've probably come across voltage regulators. A voltage regulator takes an input voltage on one pin and regulates it to a fixed voltage that is output on another pin. A third pin is used as a connection to ground. The Atari 7800 requires a 5V source to power the system, so a common 7805 linear voltage regulator was chosen by Atari.

The connection diagram for the 7805 voltage regulator used in the 7800 is shown in Figure 10.20 and Table 10.1.

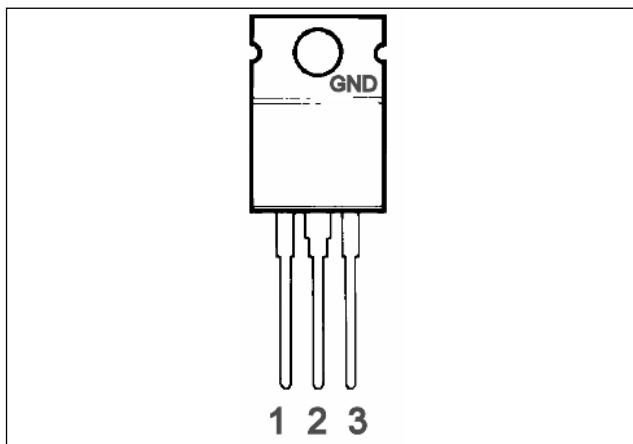
Figure 10.20 7805 Voltage Regulator Pinout

Table 10.1 Voltage Regulator Pins

Pin	Function
1	Input voltage (7V-20V)
2	Ground
3	Output voltage (5V)

When looking at a voltage regulator of the same type as found in the 7800, hold it so that the markings printed on the front of the device are facing you. The left pin is the input voltage, the middle pin (as well as the mounting point for the heat sink) is ground, and the right pin is the output voltage.

To function properly, linear voltage regulators must be fed a higher voltage than is expected on output. The 7805 requires a minimum input voltage of about 7V to produce a reliable 5V output. Linear regulators are notoriously inefficient; most of the unused energy is lost as heat, which is why voltage regulators often have a heat sink attached. Without the heat sink, the voltage regulator might overheat and could malfunction.

Power Supply Plug Retrofit

The Atari 7800 uses a proprietary power connector instead of a standard power connector typically used on electronic devices (as well as all of Atari's other game consoles). Due to this proprietary connector, it can often be difficult to find a replacement power supply, because the console has not been on the market in over a decade. The plug attached to the power cord is shown in Figure 10.21, and the connector it plugs into is shown in Figure 10.22.

We'll never know the logic behind Atari's decision to use this odd connector (were they maybe worried about losing sales of replacement power supplies to third-party companies?), but at least we can easily rectify the situation. This hack will show you how to add a standard power jack to your Atari 7800 so you can use an off-the-shelf power adapter. We'll leave the existing 7800 power jack in place so you can continue to use it if you choose.

Figure 10.21 Atari 7800 Power Connector

Figure 10.22 Atari 7800 Power Receptacle

Preparing for the Hack

For this hack you'll need:

- A panel-mount, 1/8-inch phone jack (Radio Shack part #274-248)
- Two 12-inch lengths of 18AWG insulated copper wire (preferably red and black, to connect the phone jack to the 7800 power connections)
- A 5V, 1A DC power supply with a 1/8-inch connector, center positive (to use with your new power jack, once assembled)

You can also use an Atari 2600 power supply, which is more common, though it supplies only 500mA of power. This is sufficient to power the 7800, but some cartridges, such as the Cuttle Cart, might not operate properly.

The tools required for this hack are:

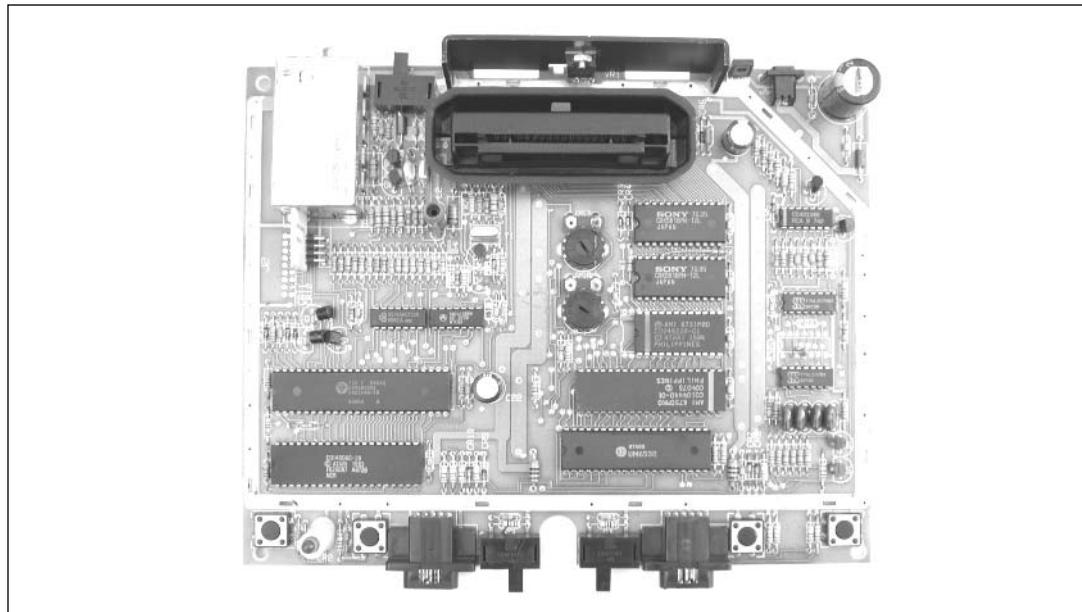
- A Phillips head screwdriver, standard size
- Wire cutters and stripper
- A soldering iron
- A small adjustable wrench or needlenose pliers
- A heat gun
- A drill or Dremel tool

Performing the Hack

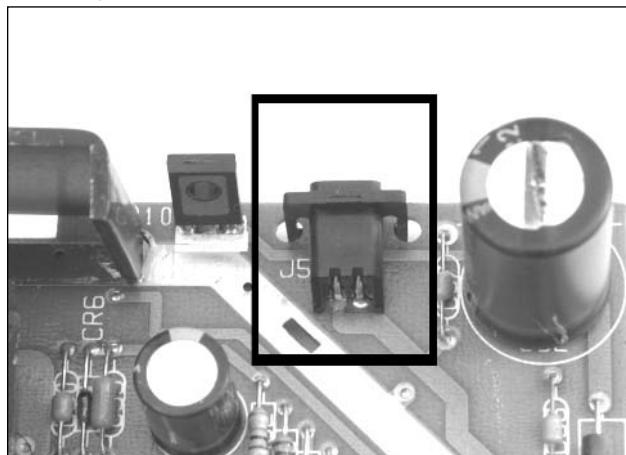
Perform the following:

1. First open the 7800 and remove the RF shield as described in the blue LED modification hack earlier in this chapter. This will give you access to the 7800 circuit board, which should resemble the one shown in Figure 10.23.

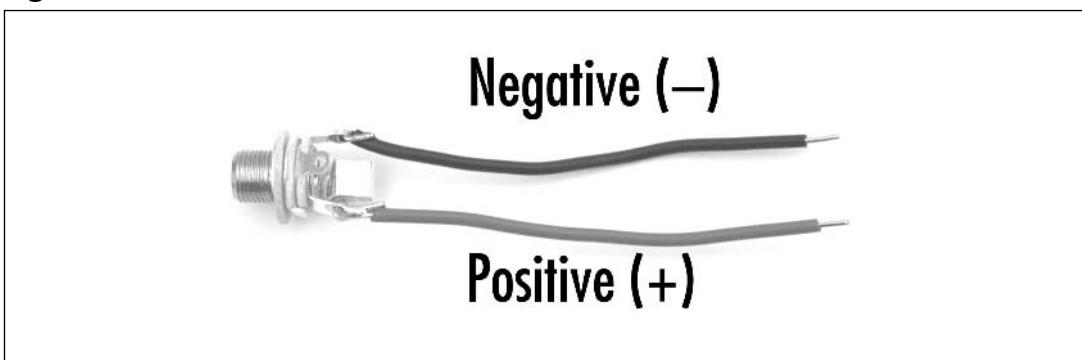
Figure 10.23 Atari 7800 Circuit Board



2. The existing power supply connector is on the upper-right corner of the board, adjacent to a large capacitor, as shown in Figure 10.24. We'll be leaving the existing connector in place, but an alternative to this hack is to remove the existing power connector and place a new connector directly onto the board. We'll mount the new connector in the bottom shell of the case, about a half-inch down from the top edge of the shell.

Figure 10.24 Existing Atari 7800 Power Connector

3. Remove the nut and washer from the 1/8-inch phone jack. These will be used to secure the jack to the case, so put them aside for now. Cut two 4-inch lengths of the 18AWG wire. You can use longer lengths of wire to give you more flexibility. These wires will connect power and ground from the 7800 circuit board to the phone jack. If you can, use black wire for the ground connection and red wire for the positive connection, as shown in Figure 10.25. Strip a short piece of insulation from the ends of each wire.

Figure 10.25 Wired Phone Jack

4. Solder one wire to the center post of the phone jack (which will be positive) and the other to the ground terminal. There could be more than two terminals, so if you're unsure which is positive and which negative, use a continuity tester to verify that you are attaching the wires to the correct locations.
5. Now it's time to solder these connections to the 7800 board. It's easiest to solder these connections to the backside of the board, so that's what we'll do here. Flip the board over and

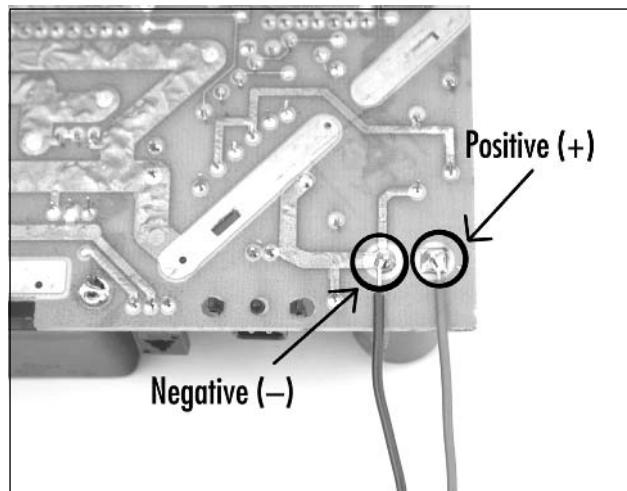
locate the corner where the existing power connector is attached. We will solder to the two large solder pads directly to the right of the connector (as shown in Figure 10.26). Solder the positive wire to the right solder pad and the ground wire to the left solder pad.

WARNING: HARDWARE HARM



Ensure that the wires are connected to the proper locations on the phone jack and the 7800 circuit board. Failure to do so could cause damage to your 7800, power supply, or circuit breakers in your home when you apply power.

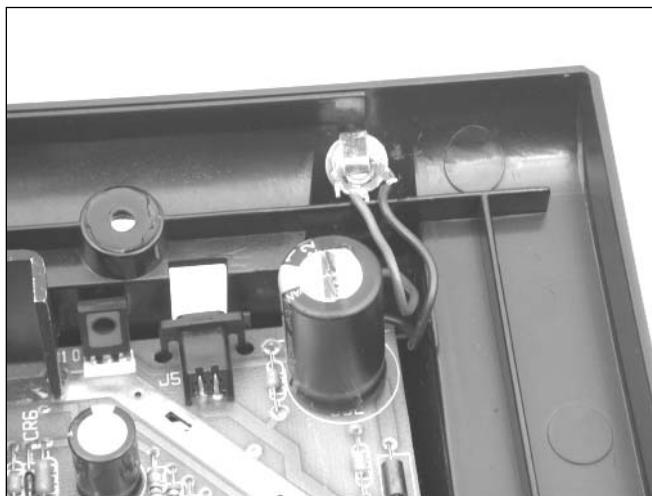
Figure 10.26 Solder Wires to the Board



6. The next step is to mount the phone jack onto the back of the 7800 housing. Using a Dremel tool or drill, drill a hole in the bottom half of the 7800 case, as indicated in Figure 10.27. The size of the opening you need to drill should be specified on the packaging for the connector. If it's not, you can eyeball the size, taking care not to drill the hole too large. It's better to err on the small size and work your way up if necessary. You may opt to place the connector elsewhere, but make sure the connector and wires will not interfere with placement of the circuit board in the case.

Figure 10.27 Drill a Hole into the Atari 7800 Case for the Phone Jack

7. Replace the RF shield back onto the 7800 circuit board, place the board back into the bottom of the 7800 case, and place the phone plug connector through the hole you just created. Thread the washer and nut onto the opposite side of the jack, tightening them so that the connector is securely attached to the 7800 case, as shown in Figure 10.28.

Figure 10.28 Attach the New Power Connector to the Case

8. Finish reassembly of the 7800 by attaching the top half of the case and then screwing the case back together with the five screws you removed at the beginning of this hack. When reassembled, your system should resemble Figure 10.29.

Figure 10.29 Back of the Atari 7800 Showing the New Power Jack



Now it's time to test your work. Hook the 7800 to a video source, insert a 2600 or 7800 game, and plug in an appropriate power supply to the new jack you just added. Turn the system on, and enjoy!

Other Hacks

The hacks presented in this chapter are only the beginning of what you can do with your Atari 7800 console. We've listed some of the other interesting, available hacks here; detailed instructions for many of these are available on the Internet. Whenever possible, we've pointed you toward resources for additional information.

Atari 7800 Composite and S-Video Output

Although the standard RF output on the Atari 7800 seems to be a bit better than that of the 2600, it can be improved further by performing a composite and/or S-Video modification. Jay Tilton (http://home.earthlink.net/~resqsoft/7800_mod.htm) and Mark Graybill (www.geocities.com/atari7800mod/7800_vidmod_construction.html) have both created video modifications for the 7800.

Sega Genesis to Atari 7800 Controller Modification

Whether through accident or design, Atari did a reasonably good job with the ergonomics of the original joystick and paddle controllers for the Atari 2600. Unfortunately, Atari's follow-on game systems, the 5200 and 7800, suffer from very uncomfortable controller designs. Although the Atari 7800 controller is more bearable than the troublesome 5200 controller, many still find its design uncomfortable to use for extended duration. With the 7800, Atari returned to an industry-standard DB-9 connector for the controller jacks, maintaining backward compatibility with 2600 controllers (since the 7800 itself is backward compatible with 2600 games). However, games that require the two dis-

tinct fire buttons of the 7800 controller cannot be enjoyed with standard 2600 controllers, which only have a single fire button.

One benefit of this backward compatibility is that Sega Genesis and Sega Master System controllers, both of which use a DB-9 connector, can be modified for use with the 7800 to support both fire buttons. Instructions on how to do so can be found in the Atari 2600 FAQ (www.atariage.com/2600/faq/index.html#seg3).

If you'd prefer to build an adapter instead of modifying existing Sega controllers, John Soper has posted schematics and instructions on how to build three different versions (www94.pair.com/jsoper/7800_gen_adap.html).

NES Control Pad to Atari 7800 Controller Modification

Along the same lines as the Sega Genesis modification previously mentioned, a Nintendo NES control pad can also be modified to work with the Atari 7800 and support both fire buttons. This hack differs from the Atari 2600 version of the NES control pad hack described in Chapter 8. Complete instructions can be found at www.atariage.com/2600/archives/nes_atari.html.

Atari 7800 DevOS Modification and Cable Creation

The Basic Input/Output System (BIOS) built into the Atari 7800 can be replaced with a Development Operating System (called the *DevOS*), which was created by Eckhard Stolberg, an avid classic gamer and master programmer. DevOS allows the Atari 7800 to be used to retrieve the contents of 7800 and 2600 cartridges to a PC via a parallel port cable. With DevOS installed, your 7800 will display a menu that allows you to transfer data from the 7800 to the PC, load data into a special RAM cartridge, or play whatever cartridge is inserted into the 7800. Complete information on how to perform this hack and build the necessary cable (and the optional RAM cartridge) can be found at Eckhard Stolberg's VCS Workshop page (<http://home.arcor.de/estolberg/tools/index.html>).

Homebrew Game Development

The older Atari 2600 and 5200 consoles have seen a wealth of homebrew development over the years, but Atari's 7800 has enjoyed very little activity on this front. There are several reasons for this; among the most compelling is that Atari used a cartridge validation key to prevent third-party companies from releasing games for the 7800 without properly licensing the rights from Atari (similar to Nintendo's attempt to do the same with its NES system, as described in Chapter 7). Atari was unable to prevent third parties from making games for its 2600 and 5200 systems but finally came up with this method to prevent unlicensed distribution of games on the 7800. This validation key is also used by the system to determine if a 7800 cartridge (as opposed to a 2600 cartridge) is inserted into the cartridge slot. As a result, this validation scheme presented a hurdle to creating new, original software for the Atari 7800.

The way the validation key works is fairly simple. Once the system has been powered on with a cartridge in place, the 7800 BIOS performs a mathematical algorithm on the cartridge data and the encryption key. The BIOS uses the result of this calculation to determine if a valid 7800 cartridge has

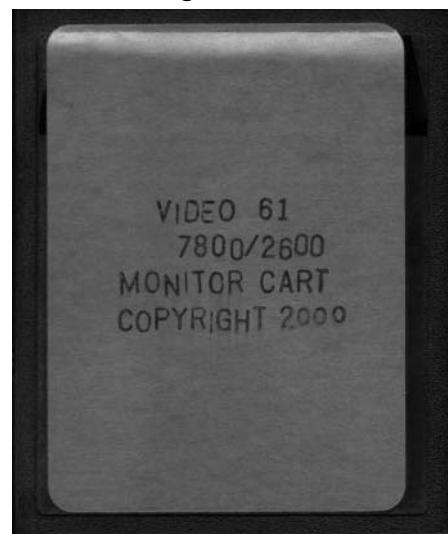
been inserted. If so, the system switches into 7800 mode; otherwise, it is assumed that a 2600 cartridge has been inserted. A 7800 cartridge that has not been properly “signed” with the proper validation key will cause the 7800 to switch into 2600 mode and thus fail to work. The validation key does not interfere with 7800 emulation, since emulators can force a valid result regardless of what 7800 software is loaded. Fortunately, the original Atari ST utility used by Atari developers to generate the validation key was released several years ago, and a PC-based utility is now available.

So, with this challenge out of the way, the floodgates to 7800 development should have opened, right? Well, not exactly. The Atari 7800 was one of Atari's least successful systems and doesn't have nearly the fan base of the Atari 2600 and 5200. The 2600, by far, dominates the homebrew scene for classic gaming systems. The Atari 5200 comes in at second place, with a reasonably active homebrew scene. As this book goes to press, the 7800 has yet to see an original 7800 game completed by homebrew authors, though a handful of projects have been started.

In regard to development systems and tools, the Atari 7800 has an advantage over its 2600 and 5200 siblings. Unlike the Atari 2600 and 5200, Atari created a purpose-built development system for the Atari 7800. This system consists of a large board that plugs into the Atari 7800 with a cable that then connects the 7800 to an Atari ST computer. Games were developed on the Atari ST using proprietary software and uploaded to the 7800 for testing and debugging. These Atari 7800 developer kits can still be found and purchased online, but it is generally easier to write games these days using modern compilers and emulators.

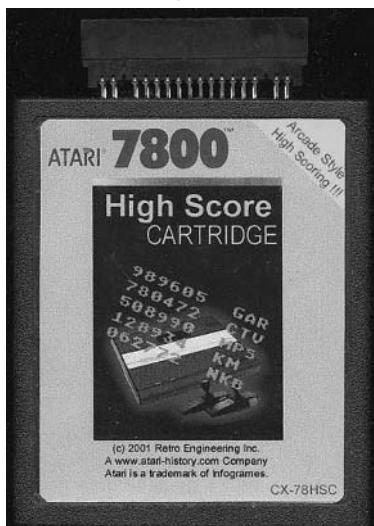
Although no new homebrew games have yet been completed for the 7800, there have been several utility and development cartridges released. The first cartridge to appear in the community was the Video 61 (formerly Harry Dodgson) 7800/2600 Monitor Cartridge (see Figure 10.30). This cartridge plugs into the 7800 and allows you to develop and debug software directly on the console. Through the use of a serial cable and terminal software, you can write software on a PC or Atari ST and then upload your work to the 7800.

Figure 10.30 7800/2600 Monitor Cartridge



Another interesting cartridge for the Atari 7800 is the Atari 7800 High Score Cart (see Figure 10.31). This cartridge was designed by Atari and intended for release with the system in 1984, but the project was canceled. The High Score Cart plugs into the 7800's cartridge slot, and Atari 7800 games are then plugged in on top of it. Games would access routines in the High Score Cart to save and load high scores. Nine of the original Atari 7800 games were written to take advantage of this cartridge. In 1999, Curt Vendel of the Atari History Museum received a copy of the original schematic and firmware for the High Score Cart from a former Atari engineer, Gary Rubio. Curt produced a working version of the High Score Cart and made a limited quantity that were sold to Atari 7800 enthusiasts.

Figure 10.31 Atari 7800 High Score Cartridge



One of the more recent 7800 developments is the Cuttle Cart 2, sequel to the popular Cuttle Cart. The original Cuttle Cart was designed for the Atari 2600 and uses an audio interface to load Atari 2600 game binaries into its on-board memory (using the same method employed by the once-popular Starpath Supercharger accessory). However, only one game can be loaded at a time, and the contents of the Cuttle Cart's memory is lost when you turn off the Atari 2600. Nonetheless, it is an extremely popular cartridge for 2600 enthusiasts, but it is no longer being produced (over 200 were manufactured and sold at a cost of \$100 each).

Chad Schell of Schell's Electronics, creator of the Cuttle Cart, recently designed a significantly enhanced version for the Atari 7800. The Cuttle Cart 2 (see Figure 10.32) allows you to copy a large number of 2600 and 7800 games onto a Multimedia Card (MMC) using your home computer. The MMC then plugs into the top of the Cuttle Cart 2. When you power up your 7800 with the Cuttle Cart 2 plugged in, you are presented with a sophisticated menu system that allows you to choose games to play, view game manuals, modify various settings, and even load game binaries through a serial interface (which is excellent for game development and debugging). As with the original Cuttle

Cart, the Cuttle Cart 2 is a great tool for homebrew developers. As of this writing, the original run of the Cuttle Cart 2 has sold out, but Chad Schell is collecting names for those who are interested in a second run.

Figure 10.32 Cuttle Cart 2 by Schell's Electronics



It's our hope that products like the Cuttle Cart 2, as well as the 7800 developer documentation that is slowly coming to light, will help spur homebrew development for this system. In most regards, the 7800's technical capabilities outclass those of the 2600 and 5200, so exciting homebrew games are possible, and we expect to see great things in the future. If you'd like to learn more about the technical aspects of the 7800, the following links will get you started:

- **Eckhard Stolberg's VCS Workshop, <http://home.arcor.de/estolberg>** Eckhard Stolberg has put together an excellent selection of various tools and documentation for Atari 2600 and 7800 programmers. Highlights on the site include information on the Atari 7800 DevOS and the 7800 Signature Key Generator to bypass the signature check in the Atari 7800 (which is a must-have for anyone writing new 7800 games).
- **Dan B's Atari 7800 Tech Page, <http://atarihq.com/danb/a7800.shtml>** Dan Boris has assembled a wide assortment of technical information, tools, and demo source code for several Atari platforms, including the Atari 7800. Here you'll find information on the 7800 system specifications, cartridge port details, information on the 7800 signature key, various technical and development documentation, and more.
- **Schell's Electronics, www.schells.com** Schell's Electronics created the Cuttle Cart for the Atari 2600 and the Cuttle Cart 2 for the Atari 7800. This cartridge is highly recom-

mended if you're a fan of the 7800 and if you want to try your hand at homebrew development. The site provides technical information and support for the Cuttle Cart products.

- **Multiple Emulator Super System, www.mess.org** The Multiple Emulator Super System (MESS) emulator supports a wide variety of classic game consoles and computers, including the Atari 7800. At this time MESS is the best Atari 7800 emulator available and is a must-have tool for anyone developing Atari 7800 software.

Atari 7800 Resources on the Web

Even though the 7800 didn't have the success of the Atari 2600 in the marketplace, it still has a strong following. Many Web sites focus entirely on the Atari 7800, and many others present a significant amount of 7800-related material. The following are our favorite Web sites that cover the 7800, but you can always locate more through the magic of your favorite Internet search engine:

- **AtariAge, www.atariage.com** Evolved from The Atari 2600 Nexus in 2001, AtariAge contains a wide variety of information about Atari game systems and a wealth of 7800-specific information. This includes a database of all released 7800 games, cartridge, manual, and box scans, screenshots, catalog scans, information on 7800 emulation, 7800 technical information, and much more. Additionally, AtariAge sports a 7800 forum where you can discuss the 7800 with other fans. The Homebrew Forums on AtariAge are a great source of information about homebrew development and are frequented by many homebrew authors writing games for various Atari systems.
- **Atari History Museum, www.atarimuseum.com** The Atari History Museum is a comprehensive site covering the entire range of Atari's history and has an extensive section on the Atari 7800, with great historical insight into the 7800, pictures, original Atari documentation, and more. Curt Vendel, the museum's curator, has been posting new Atari 7800 documentation (such as source code from original 7800 games) to the 7800 Forum on AtariAge, and we expect these documents will eventually find a permanent home on the Atari History Museum site.
- **Institute for Advanced Atari Gaming Studies, www.atari7800.com** Atari7800.com is dedicated to the Atari 7800 ProSystem and contains one of the largest storehouses of 7800 information on the Web. Here you can learn about the 7800's history, read about all the games released for the system, view the accessories Atari released, download 7800 software, desktop backgrounds, screensavers, and much more.
- **The Atari 7800 Page, www.atari7800.org** Run by Atari 7800 fan Mitchell Orman (who also helps moderate the 7800 Forum on AtariAge), Atari7800.org is one of the oldest 7800 sites on the Web, containing a large assortment of 7800 information.
- **AtariProtos.com, www.AtariProtos.com** Devoted to unearthing the secrets of Atari prototypes, AtariProtos.com features a growing repository of comprehensive prototype

reviews for the 2600, 5200, and 7800. Matt Reichert has posted insightful reviews of several Atari 7800 prototypes, including games that were never commercially released.

- **B&C ComputerVisions, www.myatari.com** B&C ComputerVisions has been serving the Atari videogaming and Atari computing communities for nearly 20 years. B&C stocks over 5000 Atari-related products and accepts and ships orders for Atari computers, videogames, and parts worldwide. The company also services most Atari products, except for monitors, power supplies, and coin-operated Atari games.
- **Best Electronics, www.best-electronics-ca.com** Best Electronics carries a large range of replacement parts and accessories for Atari game systems and computers and has served Atari users for 20 years. Featuring one of the largest inventories of Atari parts, Best Electronics has a printed catalog of more than 200 pages containing over 4000 Atari products, parts, and accessories.

Electrical Engineering Basics

Topics in this Appendix:

- Introduction
- Fundamentals
- Basic Device Theory
- Soldering Techniques
- Common Engineering Mistakes
- Web Links and Other Resources

Note: Not all hacks in this book require electrical engineering.

Introduction

Understanding how hardware hacks work usually requires an introductory-level understanding of electronics. This appendix describes electronics fundamentals and the basic theory of the most common electronic components. We also look at how to read schematic diagrams, how to identify components, proper soldering techniques, and other engineering topics.

NEED TO KNOW...LIMITATIONS OF THIS APPENDIX



Engineering, like hardware hacking, is a skill that requires time and determination if you want to be proficient in the field. There is a lot to discuss, but we have a limited amount of space. This appendix is not going to turn you into an electronics guru, but it will teach you enough about the basics so that you can start to find your way around. For more detail on the subject, see the suggested reading list at the end of this appendix.

Fundamentals

It is important to understand the core fundamentals of electronics before you venture into the details of specific components. This section provides a background on numbering systems, notation, and basic theory used in all facets of engineering.

Bits, Bytes, and Nibbles

At the lowest level, electronic circuits and computers store information in binary format, which is a base-2 numbering system containing only 0 and 1, each known as a *bit* (derived from a combination of the words *binary*, which is defined as something having two parts or components, and *digit*). The common decimal numbering system that we use in everyday life is a base-10 system, which consists of the digits 0 through 9.

Electrically, a 1 bit is generally represented by a positive voltage (5V, for example), and a 0 bit is generally represented by a zero voltage (or ground potential). However, many protocols and definitions map the binary values in different ways.

A group of 4 bits is a *nibble* (also known as a *nybble*), a group of 8 bits is a *byte*, and a group of 16 bits is typically defined as a *word* (though a *word* is sometimes defined differently, depending on the system architecture you are referring to). Figure A.1 shows the interaction of bits, nibbles, bytes, and words. This visual diagram makes it easy to grasp the concept of how they all fit together.

Figure A.1 Breakdown of a 16-Bit Word into Bytes, Nibbles, and Bits

Word															
Byte 1 (High)								Byte 0 (Low)							
Nibble 3				Nibble 2				Nibble 1				Nibble 0			
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1	0	1	1	0	1	1	0	1	0	1	0	1	0	1	0

The larger the group of bits, the more information that can be represented. A single bit can represent only two combinations (0 or 1). A nibble can represent 2^4 (or 16) possible combinations (0 to 15 in decimal), a byte can represent 2^8 (or 256) possible combinations (0 to 255 in decimal), and a word can represent 2^{16} (or 65,536) possible combinations (0 to 65,535 in decimal).

Hexadecimal format, also called *hex*, is commonly used in the digital computing world to represent groups of binary digits. It is a base-16 system in which 16 sequential numbers are used as base units before adding a new position for the next number (digits 0 through 9 and letters A through F). One hex digit can represent the arrangement of 4 bits (a nibble). Two hex digits can represent 8 bits (a byte). Table A.1 shows equivalent number values in the decimal, hexadecimal, and binary number systems. Hex digits are sometimes prefixed with 0x or \$ to avoid confusion with other numbering systems.

Table A.1 Number System Equivalents: Decimal, Binary, and Hexadecimal

Decimal	Binary	Hex
0	0	0
1	1	1
2	10	2
3	11	3
4	100	4
5	101	5
6	110	6
7	111	7
8	1000	8
9	1001	9

Continued

Table A.1 Number System Equivalents: Decimal, Binary, and Hexadecimal

Decimal	Binary	Hex
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F
16	10000	10
17	10001	11
18	10010	12
19	10011	13
20	10100	14
21	10101	15
22	10110	16
23	10111	17
24	11000	18
25	11001	19
26	11010	1A
27	11011	1B
28	11100	1C
29	11101	1D
30	11110	1E
31	11111	1F
32	100000	20
...
63	111111	3F
...
127	1111111	7F
...
255	11111111	FF

The American Standard Code for Information Interchange, or ASCII (pronounced *ask-key*), is the common code for storing characters in a computer system. The standard ASCII character set (see Table A.2) uses 1 byte to correspond to each of 128 different letters, numbers, punctuation marks, and special characters. Many of the special characters are holdovers from the original specification created in 1968 and are no longer commonly used for their originally intended purpose. Only the decimal

values 0 through 127 are assigned, which is half of the space available in a byte. An extended ASCII character set uses the full range of 256 characters available in a byte. The decimal values of 128 through 255 are assigned to represent other special characters that are used in foreign languages, graphics, and mathematics.

Table A.2 The Standard ASCII Character Set

Hex	Symbol	Hex	Symbol	Hex	Symbol	Hex	Symbol
0x00	NUL (null)	0x20	SP (space)	0x40	@	0x60	' (Single quote)
0x01	SOH (start of heading)	0x21	!	0x41	A	0x61	a
0x02	STX (start of text)	0x22	"	0x42	B	0x62	b
0x03	ETX (end of text)	0x23	#	0x43	C	0x63	c
0x04	EOT (end of transmission)	0x24	\$	0x44	D	0x64	d
0x05	ENQ (enquiry)	0x25	%	0x45	E	0x65	e
0x06	ACK (acknowledge)	0x26	&	0x46	F	0x66	f
0x07	BEL (bell)	0x27	' (apostrophe)	0x47	G	0x67	g
0x08	BS (backspace)	0x28	(0x48	H	0x68	h
0x09	HT (horizontal tab)	0x29)	0x49	I	0x69	i
0x0A	LF (line feed/new line)	0x2A	*	0x4A	J	0x6A	j
0x0B	VT (vertical tab)	0x2B	+	0x4B	K	0x6B	k
0x0C	FF (form feed)	0x2C	, (comma)	0x4C	L	0x6C	l
0x0D	CR (carriage return)	0x2D	-	0x4D	M	0x6D	m
0x0E	SO (shift out)	0x2E	. (period)	0x4E	N	0x6E	n
0x0F	SI (shift in)	0x2F	/	0x4F	O	0x6F	o
0x10	DLE (data link escape)	0x30	0	0x50	P	0x70	p
0x11	DC1 (device control 1)	0x31	1	0x51	Q	0x71	q
0x12	DC2 (device control 2)	0x32	2	0x52	R	0x72	r
0x13	DC3 (device control 3)	0x33	3	0x53	S	0x73	s
0x14	DC4 (device control 4)	0x34	4	0x54	T	0x74	t
0x15	NAK (negative acknowledge)	0x35	5	0x55	U	0x75	u
0x16	SYN (synchronous idle)	0x36	6	0x56	V	0x76	v
0x17	ETB (end of transmission block)	0x37	7	0x57	W	0x77	w
0x18	CAN (cancel)	0x38	8	0x58	X	0x78	x
0x19	EM (end of medium)	0x39	9	0x59	Y	0x79	y
0x1A	SUB (substitute)	0x3A	: (colon)	0x5A	Z	0x7A	z

Continued

Table A.2 The Standard ASCII Character Set

Hex	Symbol	Hex	Symbol	Hex	Symbol	Hex	Symbol
0x1B	ESC (escape)	0x3B	;	0x5B	[0x7B	{
0x1C	FS (file separator)	0x3C	<	0x5C	\	0x7C	
0x1D	GS (group separator)	0x3D	=	0x5D]	0x7D	}
0x1E	RS (record separator)	0x3E	>	0x5E	^	0x7E	~
0x1F	US (unit separator)	0x3F	?	0x5F		0x7F	Del (delete) (underscore)

Reading Schematics

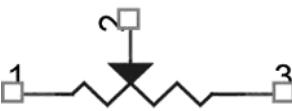
Before we get into the theory of individual electronic components, it is important to learn how circuit designs are drawn and described. A *schematic* is essentially an electrical road map of a circuit.

Reading basic schematics is a good skill to have, even if it is just to identify a particular component that needs to be removed during a hack. Reading schematics is much easier than it may appear, and with practice it will become second nature.

On a schematic, each component of the circuit is assigned its own symbol, unique to the type of device that it is. The United States and Europe sometimes use different symbols, and there are even multiple symbols to represent one type of part. A resistor has its own special symbol, as does a capacitor, a diode, or an integrated circuit. Think of schematic symbols as an alphabet for electronics. Table A.3 shows a selection of basic components and their corresponding designators and schematic symbols. This is by no means a complete list, and, as mentioned, a particular component type may have additional symbols that aren't shown here.

A *part designator* is also assigned to each component and is used to distinguish between two parts of the same type and value. The designator is usually an alphanumeric character followed by a unique numerical value (R1, C4, or SW2, for example). The part designator and schematic symbol are used as a pair to define each discrete component of the circuit design.

Table A.3 Designator and Schematic Symbols for Basic Electronic Components

Component	Designator	Symbol
Resistor	R	
Potentiometer (variable resistor)	R	
Capacitor (nonpolarized)	C	

Continued

Table A.3 Designator and Schematic Symbols for Basic Electronic Components

Component	Designator	Symbol
Capacitor (polarized)	C	
Diode	D	
LED	D	
Photodiode	D	
Transistor (NPN)	Q	
Transistor (PNP)	Q	
Crystal	Y	
Switch	SW	
Pushbutton switch	SW	
Speaker	LS	
Fuse	F	
Battery	BT	

Continued

Table A.3 Designator and Schematic Symbols for Basic Electronic Components

Component	Designator	Symbol
	None	
Ground	None	
	None	

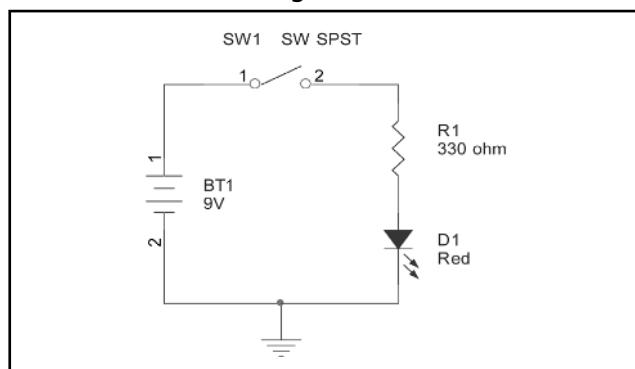
Figure A.2 An Example Circuit: A Basic LED with a Current-Limiting Resistor and Switch

Figure A.2 shows an example circuit using some of the basic schematic symbols. It describes a light-emitting diode (LED) powered by a battery and controlled by a switch. When the switch is off, no current is able to flow from the battery through the rest of the circuit, so the LED will not illuminate. When the switch is enabled, current will flow and the LED will illuminate.

Voltage, Current, and Resistance

Voltage and current are the two staple quantities of electronics. *Voltage*, also known as a *potential difference*, is the amount of work (energy) required to move a positive charge from a lower potential (a more negative point in a circuit) to higher potential (a more positive point in a circuit). Voltage can be thought of as an electrical pressure or force and has a unit of volts (V). It is denoted with a symbol V , or sometimes E or U .

Current is the rate of flow (the quantity of electrons) passing through a given point. Current has a unit of amperes, or *amps* (A), and is denoted with a symbol of I . Kirchhoff's Current Law states that the sum of currents into a point equals the sum of the currents out of a point (corresponding to a conservation of charge).

Power is a “snapshot” of the amount of work being done at that particular point in time and has a unit of watts (W). One watt of power is equal to the work done in 1 second by 1 volt moving 1 coulomb of charge. Furthermore, 1 coulomb per second is equal to 1 ampere. A coulomb is equal to

6.25×10^{18} electrons (a very, very large amount). Basically, the power consumed by a circuit can be calculated with the following simple formula:

$$P = V \times I$$

where

- P = Power (W)
- V = Voltage (V)
- I = Current (A)

NEED TO KNOW... DIFFERENTIATING BETWEEN VOLTAGE AND CURRENT



We use special terminology to describe voltage and current. You should refer to voltage as going *between* or *across* two points in a circuit—for example, “The voltage across the resistor is 1.7V.” You should refer to current going *through* a device or connection in a circuit—for example, “The current through the diode is 800mA.” When we’re measuring or referring to a voltage at a single given point in a circuit, it is defined with respect to ground (typically 0V).

Direct Current and Alternating Current

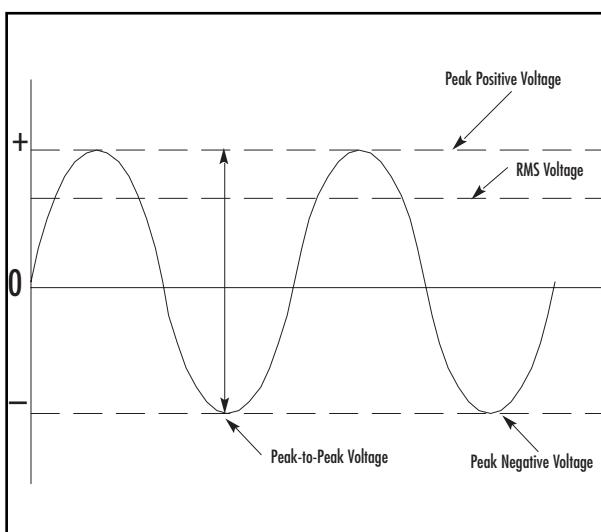
Direct current (DC) is simple to describe because it flows in one direction through a conductor and is either a steady signal or pulses. The most familiar form of a DC supply is a battery. Generally, aside from power supply or motor circuitry, DC voltages are more commonly used in electronic circuits.

Alternating current (AC) flows in both directions through a conductor (see Figure A.3) and is arguably more difficult to analyze and work with than DC. The most familiar form of an AC supply is an electrical outlet in your home. In the United States and Canada, these outlets provide 120V AC at 60Hz (cycles per second). In other parts of the world, varying AC voltages and line frequencies are used.

Several terms are used to describe the AC signal:

- **Peak voltage (V_{PEAK})** The maximum positive and negative points of the AC signal from a center point of reference.

Figure A.3 An Example of an Alternating Current Waveform



- **Peak-to-peak voltage (V_{PP})** The total voltage swing from the most positive to the most negative point of the AC signal.
- **Root-mean-square (RMS) voltage (V_{RMS})** The most common term used to describe an AC voltage. Since an AC signal is constantly changing (as opposed to DC, in which the signal is constant), the RMS measurement is the most accurate way to determine how much work will be done by an AC voltage.

For a typical sinusoidal AC signal (like the one shown in Figure A.3), the following four formulas can be used:

$$\text{Average AC Voltage } (V_{AVG}) = 0.637 \times V_{PEAK} = 0.9 \times V_{RMS}$$

$$V_{PEAK} = 1.414 \times V_{RMS} = 1.57 \times V_{AVG}$$

$$V_{RMS} = 0.707 \times V_{PEAK} = 1.11 \times V_{AVG}$$

$$V_{PP} = 2 \times V_{PEAK}$$

Resistance

Resistance can be described with a simple analogy of water flowing through a pipe: If the pipe is narrow (high resistance), the flow of water (current) will be restricted. If the pipe is large (low resistance), water (current) can flow through it more easily. If the pressure (voltage) is increased, more current will be forced through the conductor. Any current prevented from flowing (if the resistance is high, for example) will be dissipated as heat (based on the first law of thermodynamics, which states that energy cannot be created or destroyed, simply changed in form). Additionally, there will be a difference in voltage on either side of the conductor.

Resistance is an important electrical property and exists in any electrical device. *Resistors* are devices used to create a fixed value of resistance. (For more information on resistors, see the “Basic Device Theory” section in this appendix.)

Ohm’s Law

Ohm’s Law, proven in the early 19th century by George Simon Ohm, is a basic formula of electronics that states the relationship among voltage, current, and resistance in an ideal conductor. The current in a circuit is directly proportional to the applied voltage and inversely proportional to the circuit resistance. Ohm’s Law can be expressed as the following equations:

$$V = I \times R$$

Or...

$$I = V / R$$

Or...

$$R = V / I$$

Where...

- V = Voltage (V)
- I = Current (A)
- R = Resistance (in ohms, designated with the omega symbol, Ω)

Basic Device Theory

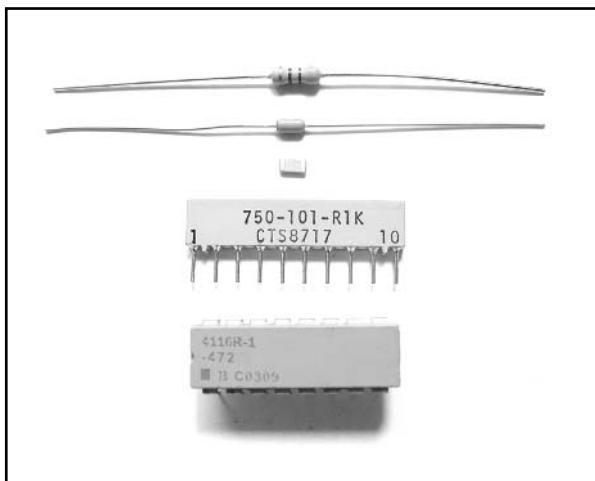
This section explores the basic device theory of the five most common electronic components: resistors, capacitors, diodes, transistors, and integrated circuits. Understanding the functionality of these parts is essential to any core electronics knowledge and will prove useful in designing or reverse-engineering products.

Resistors

Resistors are used to reduce the amount of current flowing through a point in a system. Resistors are defined by three values:

- Resistance (Ω)
- Heat dissipation (in watts, W)
- Manufacturing tolerance (%)

Figure A.4 Various Resistor Types



A sampling of various resistor types is shown in Figure A.4. Resistors are not polarized, meaning that they can be inserted in either orientation with no change in electrical function.

The value of a resistor is indicated by an industry-standard code of four or five colored bands printed directly onto the resistor (see Figure A.5). The bands define the resistance, multiplier, and manufacturing tolerance of the resistor. The manufacturing tolerance is the allowable skew of a resistor value from its ideal rated value.

A resistor's internal composition can consist of many different materials, but typically one of three are used: carbon, metal film, or wire-wound. The material is usually wrapped around a core, with the wrapping type and length corresponding to the resistor value. The carbon-filled resistor, used in most general-purpose applications such as current limiting and nonprecise circuits, allows a $\pm 5\%$ tolerance on the resistor value. Metal film resistors are for more precise applications such as amplifiers, power supplies, and sensitive analog circuitry; they usually allow a ± 1 or 2% tolerance. Wire-wound resistors can also be very accurate.

When resistors are used in series in a circuit (see Figure A.6), their resistance values are *additive*, meaning that you simply add the values of the resistors in series to obtain the total resistance. For example, if R_1 is 220 ohm and R_2 is 470 ohm, the overall resistance will be 690 ohm.

Parallel circuits provide alternative pathways for current flow, although the voltage across the components in parallel is the same. When resistors are used in parallel (see Figure A.7), a simple equation is used to calculate the overall resistance:

$$1 / R_{TOTAL} = (1 / R_1) + (1 / R_2) + \dots$$

This same formula can be extended for any number of resistors used in parallel. For example, if R_1 is 220 ohm and R_2 is 470 ohm, the overall resistance will be 149.8 ohm.

For only two resistors in parallel, an alternate formula can be used:

$$R_{TOTAL} = (R_1 \times R_2) / (R_1 + R_2)$$

Carbon and metal film resistors typically come in wattage values of $1/16W$, $1/8W$, $1/4W$, $1/2W$ and $1W$. This corresponds to how much power they can safely dissipate. The most commonly used

Figure A.5 Resistor Color Code Chart

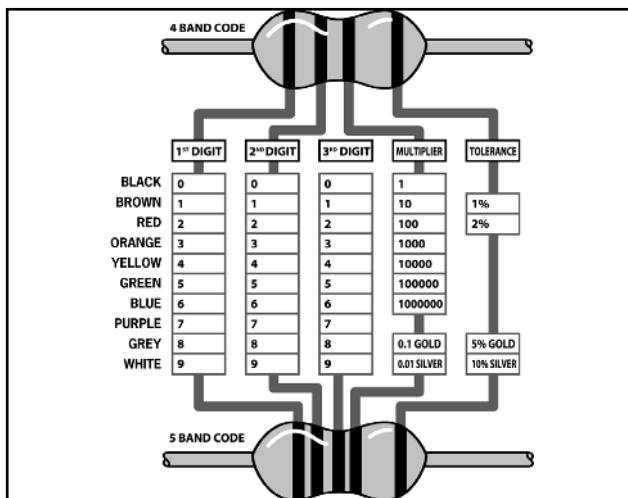


Figure A.6 Resistors in Series

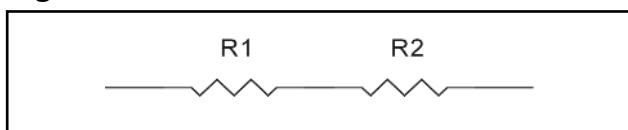
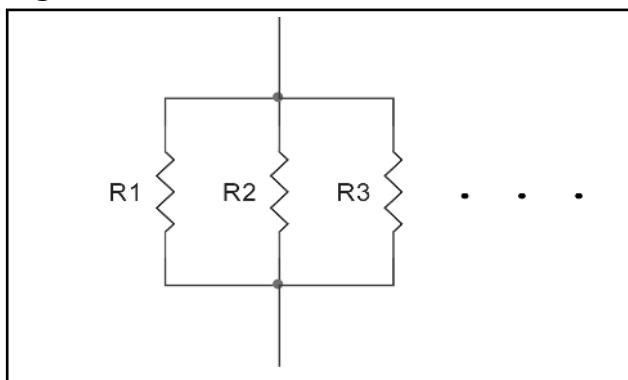


Figure A.7 Resistors in Parallel



resistors are 1/4W and 1/2W. For large current applications, wire-wound resistors are typically used because they can support wattages greater than 1W. The wattage of the resistor usually corresponds to its physical size and surface area. For most consumer electronics, resistors greater than 1W are typically not used. To calculate the required wattage value for your application, use the following equation:

$$P = V \times I$$

Or...

$$P = I^2 \times R$$

Where...

- P = Power (W)
- V = Voltage across the resistor (V)
- I = Current flowing through the resistor (A)
- R = Resistance value (Ω)

Capacitors

A *capacitor's* primary function is to store electrical energy in the form of electrostatic charge. Consider a simple example of a water tower, which stores water (charge): When the water system (circuit) produces more water than a town or building needs, the excess is stored in the water tower (capacitor). At times of high demand, when additional water is needed, the excess water (charge) flows out of the water tower to keep the pressure up.

A capacitor is usually implemented for one of three uses:

- **To store a charge** Typically used for high-speed or high-power applications, such as a laser or a camera flash. The capacitor will be fully charged by the circuit in a fixed length of time, and then all of its stored energy will be released and used almost instantaneously, just like the water tower example previously described.
- **To block DC voltage** If a DC voltage source is connected in series to a capacitor, the capacitor will instantaneously charge and no DC voltage will pass into the rest of the circuit. However, an AC signal flows through a capacitor unimpeded because the capacitor will charge and discharge as the AC fluctuates, making it appear that the alternating current is flowing.
- **To eliminate ripples** Useful for filtering, signal processing, and other analog designs. If a line carrying DC voltage has ripples or spikes in it, also known as “noise,” a capacitor can smooth or “clean” the voltage to a more steady value by absorbing the peaks and filling in the valleys of the noisy DC signal.

Capacitors are constructed of two metal plates separated by a *dielectric*. The dielectric is any material that does not conduct electricity, and varies for different types of capacitors. It prevents the plates

from touching each other. Electrons are stored on one plate of the capacitor and they discharge through the other. Consider lightning in the sky as a real-world example of a capacitor: One plate is formed by the clouds, the other plate is formed by the earth's ground, and the dielectric is the air in between. The lightning is the charge releasing between the two plates.

Depending on their construction, capacitors are either *polarized*, meaning that they exhibit varying characteristics based on the direction they are used in a circuit, or *nonpolarized*, meaning that they can be inserted in either orientation with no change in electrical function. A sampling of various capacitor types is shown in Figures A.8 and A.9.

Figure A.8 Various Nonpolarized Capacitor Types (Ceramic Disc and Multilayer)

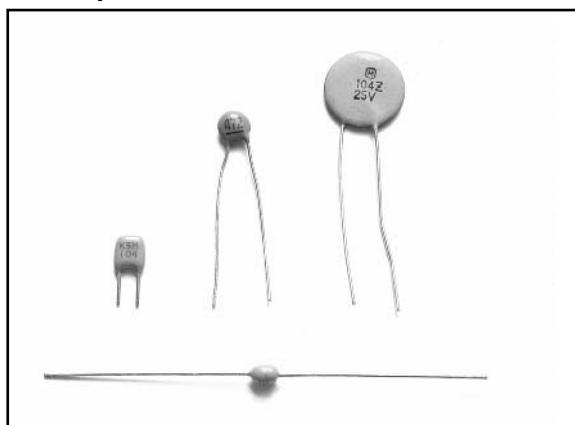
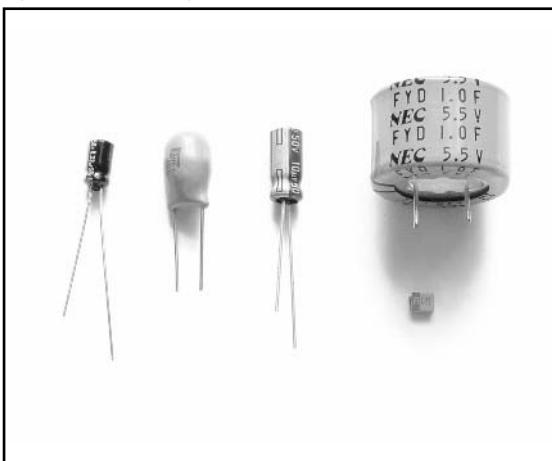
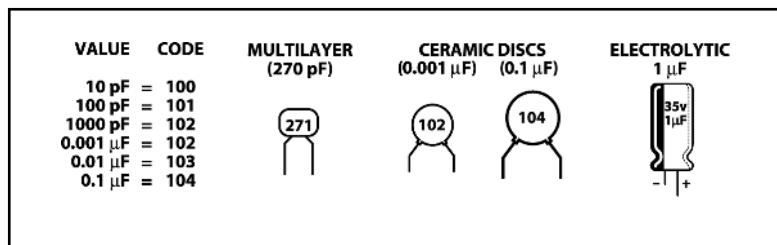


Figure A.9 Various Polarized Capacitor Types (Electrolytic and Tantalum)



Capacitors have a unit of farad (F). A 1 farad capacitor can store 1 coulomb of charge at 1 volt (equal to 1 amp-second of electrons at 1 volt). A single farad is a very large amount. Most capacitors store a minuscule amount of charge and are usually denoted in uF (microfarads, $10^{-6} \times F$) or pF (picofarads, $10^{-12} \times F$). The physical size of the capacitor is usually related to the dielectric material and the amount of charge that the capacitor can hold.

Unlike resistors, capacitors do not use a color code for value identification. Today, most monolithic and ceramic capacitors are marked with a three-number code called an *IEC marking* (see Figure A.10). The first two digits of the code indicate a numerical value; the last digit indicates a multiplier. Electrolytic capacitors are always marked in uF. These devices are polarized and must be oriented correctly during installation. Polarized devices have a visible marking denoting the negative side of the device (in the case of surface-mount capacitors, the marking is on the positive side). There may be additional markings on the capacitor (sometimes just a single character); these usually denote the capacitor's voltage rating or manufacturer.

Figure A.10 Examples of Some Capacitor IEC Markings

$$1 / C_{\text{TOTAL}} = (1 / C_1) + (1 / C_2) + \dots$$

This same formula can be extended for any number of capacitors used in series. For example, if C_1 is 100uF and C_2 is 47uF, the overall capacitance will be 31.9uF.

For only two capacitors in series, an alternate formula can be used:

$$C_{\text{TOTAL}} = (C_1 \times C_2) / (C_1 + C_2)$$

When using capacitors in series, you store effectively less charge than you would by using either one alone in the circuit. The advantage to capacitors in series is that it increases the maximum working voltage of the devices.

When capacitors are used in parallel in a circuit (see Figure A.12), their effective capacitance is additive, meaning that you simply add the values of the capacitors in parallel to obtain the total capacitance. For example, if C_1 is 100uF and C_2 is 47uF, the overall capacitance will be 147uF.

Capacitors are often used in combination with resistors in order to control their charge and discharge time. Resistance directly affects the time required to charge or discharge a capacitor (the larger the resistance, the longer the time).

Figure A.13 shows a simple RC circuit. The capacitor will charge as shown by the curve in Figure A.14. The amount of time for the capacitor to become fully charged in an RC circuit depends on the values of the capacitor and resistor in the circuit.

The calculations to determine effective capacitance of capacitors in series and parallel are essentially the reverse of those used for resistors. When capacitors are used in series (see Figure A.11), a simple equation is used to calculate the effective capacitance:

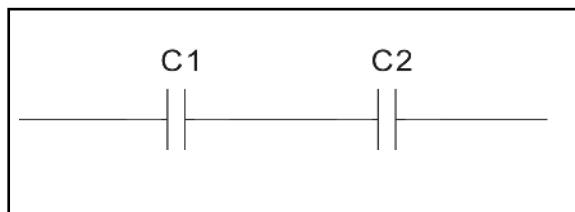
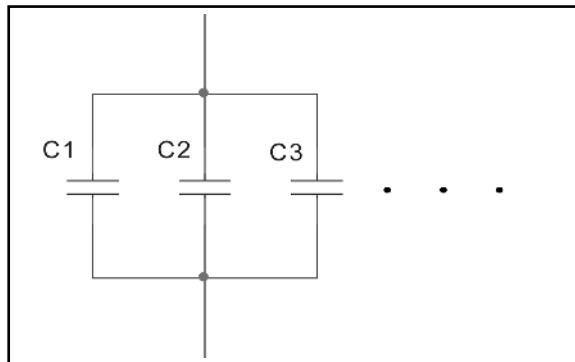
Figure A.11 Capacitors in Series**Figure A.12** Capacitors in Parallel

Figure A.13 A Simple RC Circuit to Charge a Capacitor

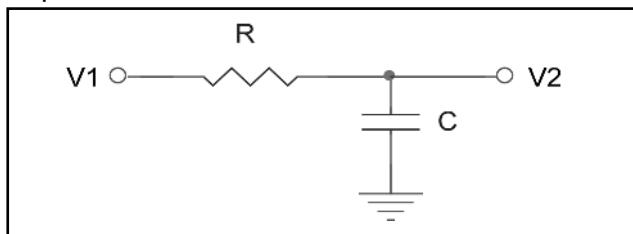


Figure A.14 Capacitor-Charging Curve

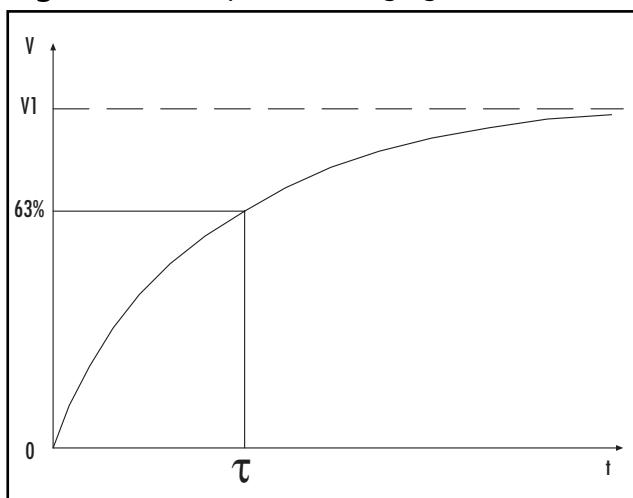
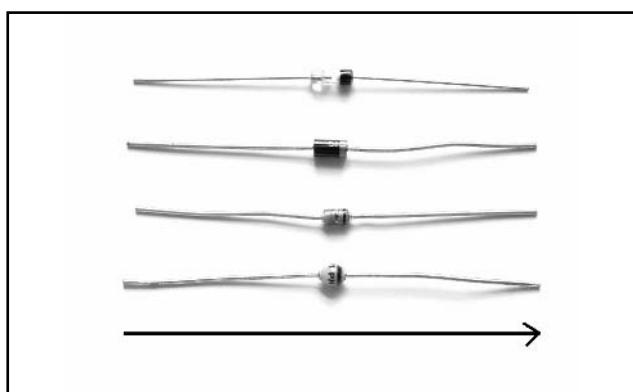


Figure A.15 Various Diode Types Showing Direction of Current Flow



The variable τ (called the *time constant*) is used to define the time it takes for the capacitor to charge to 63.2 % of its maximum capacity. The time constant can be calculated by the following formula:

$$\tau = R \times C$$

Where...

- τ = Time constant (seconds)
- C = Capacitance (F)
- R = Resistance (Ω)

A capacitor reaches 63.2 % of its charge in one-fifth of the time it takes to become fully charged. Capacitors in actual applications are usually not charged to their full capacity because it takes too long.

Diodes

In the most basic sense, *diodes* pass current in one direction while blocking it from the other. This allows for their use in rectifying AC into DC, filtering, limiting the range of a signal (known as a *diode clamp*), and as “steering diodes,” in which diodes are used to allow voltage to be applied to only one part of the circuit.

Most diodes are made with semiconductor materials such as silicon, germanium, or selenium. Diodes are polarized, meaning that they exhibit varying characteristics depending on the direction they are used in a circuit. When current is flowing through the diode in the direction shown in Figure A.15 (from anode, left, to cathode, right), the diode appears as a short circuit. When current tries to pass in the opposite direction, the diode exhibits a high resistance, preventing the current from flowing.

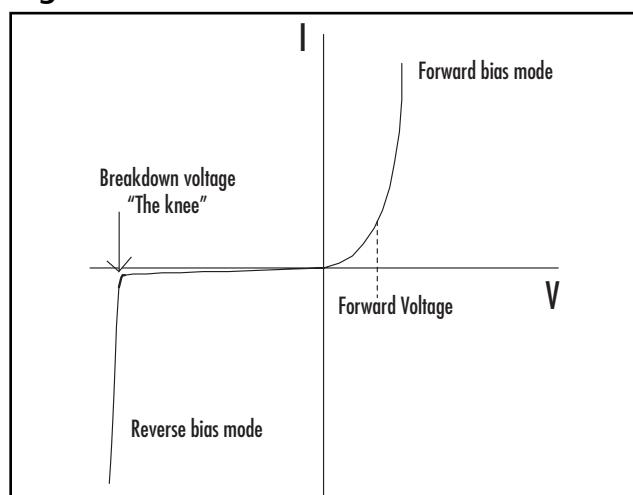
Diodes come in many types and sizes, each with varying electrical properties. You need to consider characteristics such as breakdown voltage, forward voltage, forward current, and reverse recovery time when designing with diodes or replacing one in a circuit:

- **Breakdown/reverse voltage (V_R)**, also known as the *peak inverse voltage* (P_{IV}), is the maximum voltage you can apply across a diode in the reverse direction and still have it block conduction. If this voltage is exceeded, the diode goes into “avalanche breakdown” and conducts current, essentially rendering the diode useless (unless it’s a Zener diode, which is designed to operate in this breakdown region).
- **Forward voltage (V_F)** is the voltage drop across the diode. This usually corresponds to the forward current (the greater the current flowing through the diode, the larger the voltage drop). Typical forward voltage of a general-purpose diode is between 0.5V and 0.8V at 10mA.
- **Forward current (I_F)** is the maximum current that can flow through the diode. If current flowing through the diode is more than it can handle, the diode will overheat and can melt down and cause a short circuit.
- **Reverse recovery time (T_{RR})** is the time it takes a diode to go from forward conduction to reverse blocking (think of a revolving door that goes in both directions and the people coming in and going out acting as the current). If the turnaround time is too slow, current will flow in the reverse direction when the polarity changes and cause the diode junction to heat up and possibly fail. This is primarily of concern for AC-rectifying circuits commonly used in power supplies.

Figure A.16 shows the diode V-I curve, a standard curve that describes the relationship between voltage and current with respect to a diode.

In normal forward bias operation (shown on the right side of the graph), the diode begins to conduct and act as a short circuit after the forward voltage drop is met (usually between 0.5V and 0.8V). In reverse bias operation (shown on the left side of the graph), reverse current is generally measured in the nA range (an extremely small measure of current). When the diode is reverse biased, current is essentially prevented from flowing in that direction, with the exception of a very small leakage current. The point at which the diode begins its avalanche breakdown is called “the knee,” as

Figure A.16 The Diode V-I Curve



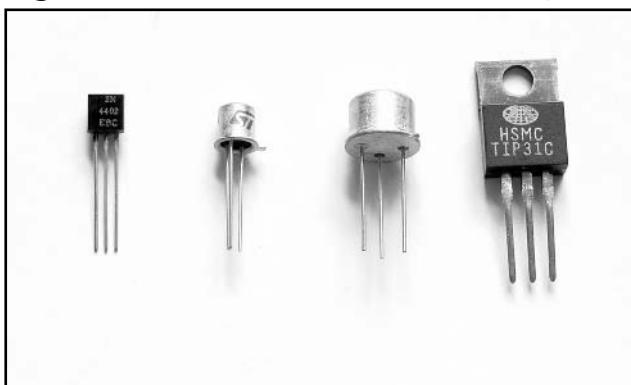
shown by the visible increase in reverse current on the curve, looking somewhat similar to a profile of a knee. Breakdown is not a desirable mode to which to subject the diode, unless the diode is of a Zener type (in which case proper current limiting should be employed).

Transistors

The *transistor* is arguably the greatest invention of the 20th century and the most important of electronic components. It is a three-terminal device that essentially serves as an amplifier or switch to control electronic current. When a small current is applied to its base, a much larger current is allowed to flow from its collector. This gives a transistor its switching behavior, since a small current can turn a larger current on and off.

The first transistor was demonstrated on December 23, 1947, by William Shockley, John Bardeen, and Walter Brattain, all scientists at the Bell Telephone Laboratories in New Jersey. The transistor was the first device designed to act as both a transmitter, converting sound waves into electronic waves, and a resistor, controlling electronic current. The name *transistor* comes from the *trans* of *transmitter* and the *sistor* of *resistor*. Although its use has gone far beyond the function that combination implies, the name remains.

Figure A.17 Various Discrete Transistor Types



This appendix only scratches the surface of transistor theory and focuses only on the most general terms. A sampling of various discrete transistors is shown in Figure A.17.

The transistor is composed of a three-layer “sandwich” of semiconductor material. Depending on how the material’s crystal structure is treated during its creation (in a process known as *doping*), it becomes more positively charged (P-type) or negatively charged (N-type). The transistor’s three-layer structure contains a P-type layer between N-type layers (known as an NPN configuration) or an N-type layer sandwiched between P-type layers (known as a PNP configuration).

The voltages at a transistor terminal (C for the collector, E for the emitter, and B for the base) are measured with respect to ground and are identified by their pin names, V_C , V_E , and V_B , respectively. The voltage drop measured between two terminals on the transistor is indicated by a double-subscript

The transistor became commercially available on May 10, 1954 from Texas Instruments, and quickly replaced the bulky and unreliable vacuum tubes, which were much larger and required more power to operate. Jumping ahead 50 years, to 2004, transistors are now an essential part of engineering, used in practically every circuit and by the millions in single integrated circuits taking up an area smaller than a fingernail. Companies such as AMD, NEC, Samsung, and Intel are pushing the envelope of transistor technology, continuing to discover new ways to develop smaller, faster, and cheaper transistors.

(for example, V_{BE} corresponds to the voltage drop from the base to the emitter). Figure A.18 shows the typical single NPN and PNP schematic symbols and notations.

A trick to help you remember which diagram corresponds to which transistor type is to think of NPN as meaning “not pointing in” (in reference to the base-emitter diode). With that said, the other transistor is obviously the PNP type.

An NPN transistor has four properties that must be met (the properties for the PNP type are the same, except the polarities are reversed):

1. The collector must be more positive than the emitter.
2. The base-emitter and base-collector circuits look like two diodes back to back (see Figure A.19). Normally the base-emitter diode is conducting (with a forward voltage drop, V_{BE} , of approximately 0.7V) and the base-collector diode is reverse-biased.
3. Each transistor has maximum values of I_C , I_B , and V_{CE} that cannot be exceeded without risk of damaging the device. Power dissipation and other limits specified in the manufacturer’s data sheet should also be obeyed.
4. The current flowing from collector to emitter (I_C) is roughly proportional to the current input to the base (I_B), shown in Figure A.20, and can be calculated with the following formula:

$$I_C = h_{FE} \times I_B$$

or

$$I_C = \beta \times I_B$$

Figure A.18 NPN (Left) and PNP (Right) Transistor Diagrams

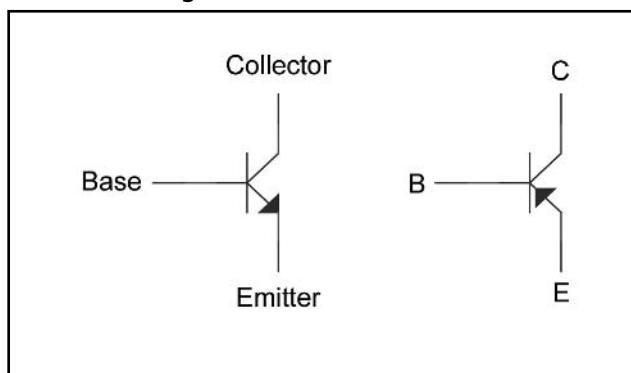


Figure A.19 Diode Representation of a Transistor, NPN (Left) and PNP (Right)

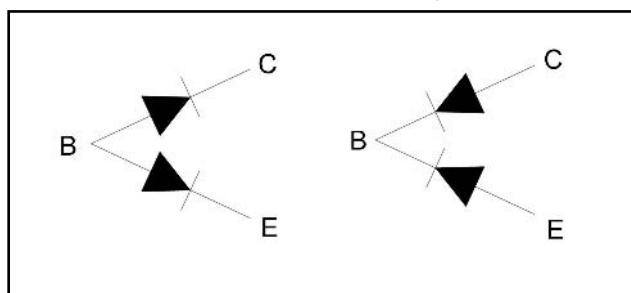
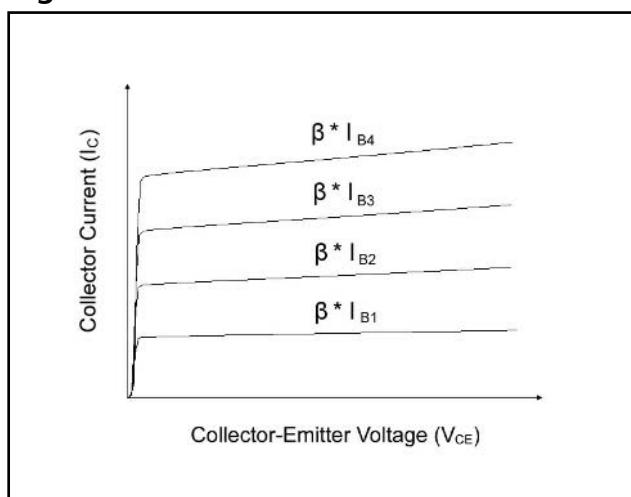


Figure A.20 NPN Transistor Characteristic



Where h_{FE} (also known as beta, β) is the current gain of the transistor. Typically, β is around 100, though it is not necessarily constant.

Integrated Circuits

Integrated circuits (ICs) combine discrete semiconductor and passive components onto a single microchip of semiconductor material. These may include transistors, diodes, resistors, capacitors, and other circuit components. Unlike discrete components, which usually perform a single function, ICs are capable of performing multiple functions. There are thousands of IC manufacturers, but some familiar ones are Intel, Motorola, and Texas Instruments.

The first generation of commercially available ICs were released by Fairchild and Texas Instruments in 1961 and contained only a few transistors. In comparison, the latest Pentium 4 processor by Intel contains over 175 million transistors in a die area of only 237mm² (approximately the size of your thumbnail).

Figure A.21 Silicon Die Inside an Integrated Circuit

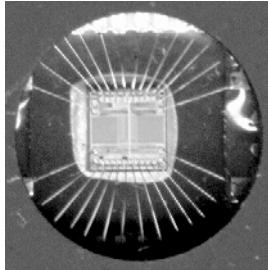
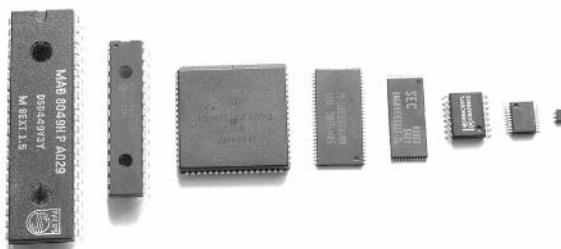


Figure A.22 Various IC Package Types



ICs are easy to identify in a circuit by their unique packaging. Typically, the silicon die (containing the microscopic circuitry) is mounted in a plastic or ceramic housing with tiny wires connected to it (see Figure A.21). The external housing (called a *package*) comes in many mechanical outlines and various pin configurations and spacings.

With the constant advances in technology, ICs are shrinking to inconceivable sizes. Figure A.22 shows a variety of IC packages, including, from left to right, Dual Inline Package (DIP), Narrow DIP, Plastic Leadless Chip Carrier (PLCC), Thin Small Outline Package (TSOP) Type II, TSOP Type I, Small Outline Integrated Circuit (SOIC), Shrink Small Outline Package (SSOP), and Small Outline Transistor (SOT-23).

Ball Grid Array (BGA) is a relatively new package type that locates all the device leads underneath the chip, which reduces the area necessary for the device (see Figure A.23). However, it is extremely difficult to access the balls of the BGA without completely removing the device, which could be a problem for hardware hacking. BGA devices are becoming more popular due to their

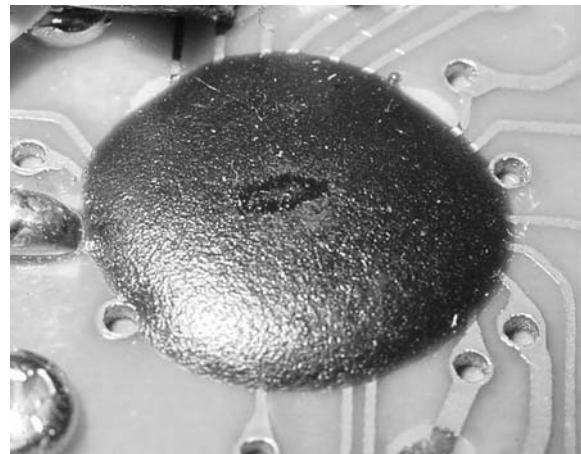
small footprint and low failure rates. The testing process (done during product manufacturing) is more expensive than other package types due to the fact that X-rays need to be used to verify that the solder has properly bonded to each of the ball leads.

With Chip-on-Board (COB) packaging, the silicon die of the IC is mounted directly to the PCB and protected by epoxy encapsulation (see Figure A.24).

Figure A.23 BGA Packaging

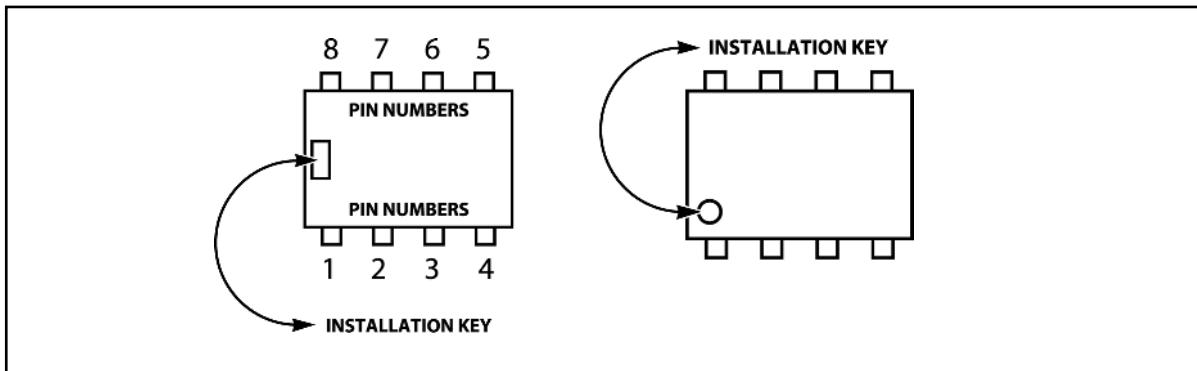


Figure A.24 COB Packaging



Proper IC positioning is indicated by a dot or square marking (known as a *key*) located on one end of the device (see Figure A.25). Some devices mark pin 1 with an angled corner (for square package types such as PLCC). On the circuit board, pin 1 is typically denoted by a square pad, whereas the rest of the IC's pads will be circular. Sometimes, a corresponding mark will be silkscreened or otherwise noted on the circuit board. Pin numbers start at the keyed end of the case and progress counter-clockwise around the device, unless noted differently in the specific product data sheet.

Figure A.25 IC Package Showing Pin Numbers and Key Marking



Microprocessors and Embedded Systems

A microprocessor – also known as a microcontroller or CPU (central processing unit), though there are slight technical differences – is essentially a general-purpose computer and is the heart of any embedded system. It is a complete computational engine fabricated on a single integrated circuit. In embedded systems, there is a union of hardware (the underlying circuitry) and software/firmware (code that is executed on the processor). You cannot have one without the other. Just about every electronic device you own can be considered an embedded system.

In November 1971, Intel released the first microprocessor, the Intel 4004. There are now thousands of microprocessors available each with their own benefits and features, including:

- Cost
- Size
- Clock speed
- Data width (for example, 8-, 16-, or 32-bit)
- On-chip peripherals (such as on-chip memory, I/O pins, LCD control, RS232/serial port, USB support, wireless support, analog-to-digital converters, or voltage references)

Common microprocessors include the Intel x86/Pentium-family (used on most personal computers), Motorola 6800- and 68000-families (such as the 68020 or 68030 used in some Macintosh computers or the DragonBall MC68328 used in some Palm PDA devices), ZiLOG Z8, Texas Instruments OMAP, and Microchip PIC.

While we don't cover the specifics of various microprocessors here, we wanted to mention their ubiquity inside hardware products. When you're hardware hacking or reverse engineering a product, chances are that you will encounter a microprocessor of some type. But fear not: Microprocessor data sheets, usually available from the manufacturer, contain instruction sets, register maps, and device-specific details that will give you the inside scoop on how to operate the device. And, once you understand the basic theory of how microprocessors work and the low-level assembly language that they execute, it is fairly trivial to apply that knowledge to a new device or processor family.

Soldering Techniques

Soldering is an art form that requires proper technique in order to be done right. With practice, you will become comfortable and experienced with it. The two key parts of soldering are good heat distribution and cleanliness of the soldering surface and component. In the most basic sense, soldering requires a soldering iron and solder. There are many shapes and sizes of tools to choose from (you can find more details in Chapter 1 “Tools of the Warranty Voiding Trade”). This section uses hands-on examples to demonstrate proper soldering and desoldering techniques.

WARNING: PERSONAL INJURY



Improper handling of the soldering iron can lead to burns or other physical injuries. Wear safety goggles and other protective clothing when working with solder tools. With temperatures hovering around 700 degrees F, the tip of the soldering iron, molten solder, and flux can quickly sear through clothing and skin. Keep all soldering equipment away from flammable materials and objects. Be sure to turn off the iron when it is not in use and store it properly in its stand.

Hands-On Example: Soldering a Resistor to a Circuit Board

This simple example shows the step-by-step process to solder a through-hole component to a printed circuit board (PCB). We use a piece of prototype PCB and a single resistor (see Figure A.26). Before

you install and solder a part, inspect the leads or pins for oxidation. If the metal surface is dull, sand with fine sandpaper until shiny. In addition, clean the oxidation and excess solder from the soldering iron tip to ensure maximum heat transfer.

Bend and insert the component leads into the desired holes on the PCB. Flip the board to the other side. Slightly bend the lead you will be soldering to prevent the component from falling out when the board is turned upside down (see Figure A.27).

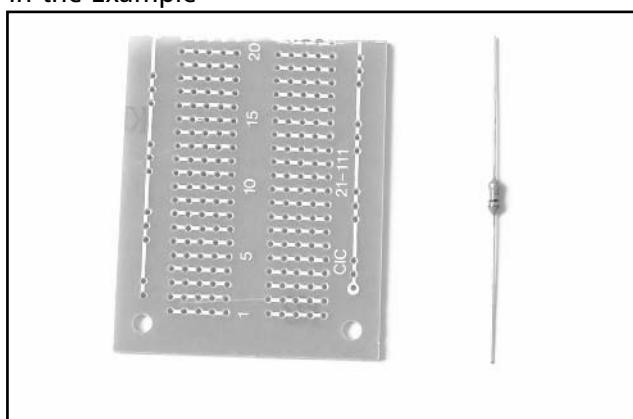
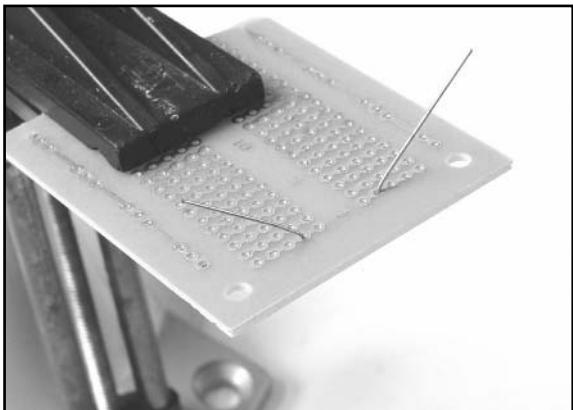
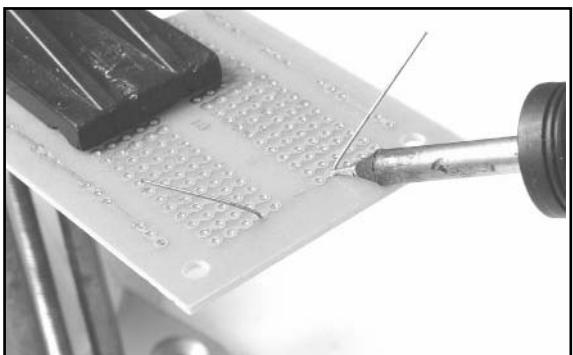
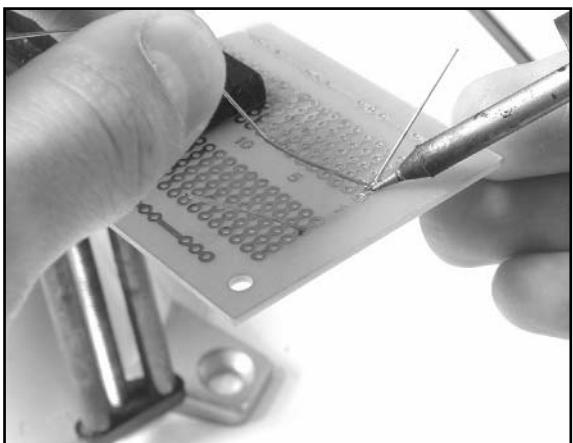


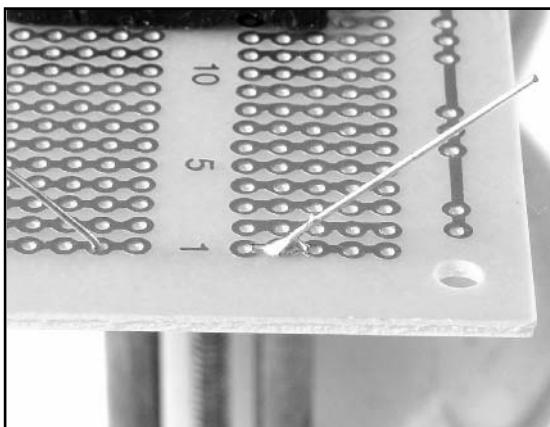
Figure A.27 Resistor Inserted into PCB**Figure A.28** Heating the Desired Solder Connection**Figure A.29** Applying Heat and Solder to the Connection

To begin the actual soldering process, allow the tip of your iron to contact both the component lead and the pad on the circuit board for about 1 second before feeding solder to the connection. This will allow the surface to become hot enough for solder to flow smoothly (see Figure A.28).

Next, apply solder sparingly and hold the iron in place until solder has evenly coated the surface (see Figure A.29). Ensure that the solder flows all around the two pieces (component lead and PCB pad) that you are fastening together. Do not put solder directly onto the hot iron tip before it has made contact with the lead or pad; doing so can cause a cold solder joint (a common mistake that can prevent your hack from working properly). Soldering is a function of heat, and if the pieces are not heated uniformly, solder may not spread as desired. A cold solder joint will loosen over time and can build up corrosion.

When it appears that the solder has flowed properly, remove the iron from the area and wait a few seconds for the solder to cool and harden. Do not attempt to move the component during this time. The solder joint should appear smooth and shiny, resembling the image in Figure A.30. If your solder joint has a dull finish, reheat the connection and add more solder if necessary.

Once the solder joint is in place, snip the lead to your desired length (see Figure A.31). Usually, you will simply cut the remaining portion of the lead that is not part of the actual solder joint (see Figure A.32). This prevents any risk of short circuits between leftover component leads on the board.

Figure A.30 Successful Solder Joint

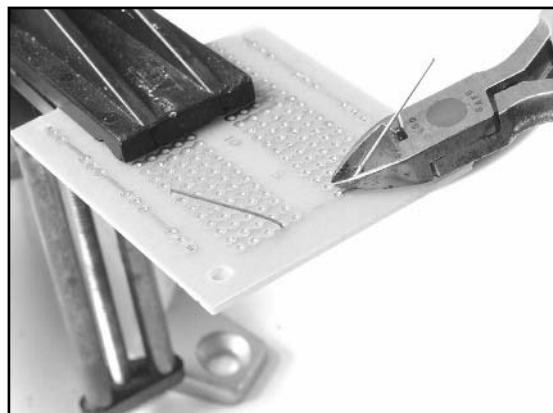
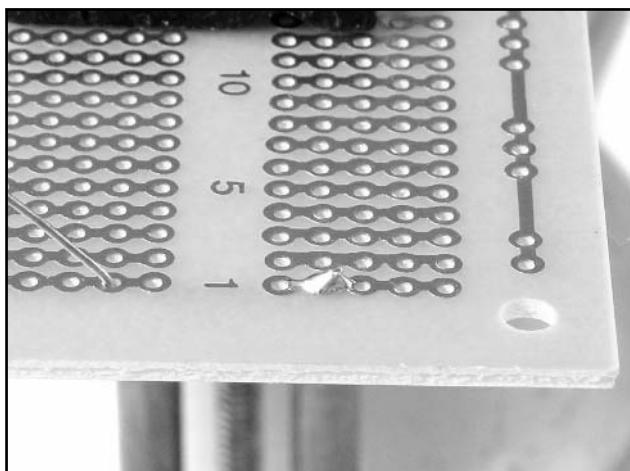
Every so often during any soldering process, use a wet sponge to lightly wipe the excess solder and burned flux from the tip of your soldering iron. This allows the tip to stay clean and heat properly. Proper maintenance of your soldering equipment will also increase its life span.

Desoldering Tips

Desoldering, or removing a soldered component from a circuit board, is typically more tricky than soldering, because you can easily damage the device, the circuit board, or surrounding components.

For standard through-hole components, first grasp the component with a pair of needle-nose pliers. Heat the pad beneath the lead you intend to extract and pull gently. The lead should come out. Repeat for the other lead. If solder fills in behind the lead as you extract it, use a spring-loaded solder sucker to remove the excess solder.

For through-hole ICs or multipin parts, use a solder sucker or desoldering braid to remove excess from the hole before attempting to extract the part. You can use a small flat-tip screwdriver or IC extraction tool to help loosen the device from the holes. Be careful to not overheat components, since they can become damaged and may fail during operation. If a component is damaged during extraction, simply replace it with a new part. For surface mount devices (SMDs) with more than a few pins, the easiest method to remove the part is by using the ChipQuik SMD Removal Kit, as shown in the following step-by-step example. Removal of SMD and BGA devices is normally accomplished with special hot-air rework stations. These stations provide a directed hot-air stream used with specific noz-

Figure A.31 Snipping Off the Remaining Component Lead**Figure A.32** Completed Soldering Example

zles, depending on the type of device to be removed. The hot air can flow freely around and under the device, allowing the device to be removed with minimal risk of overheating. Rework stations are typically priced beyond the reach of hobbyist hardware hackers, and the ChipQuik kit works quite well as a low-cost alternative.

Hands-On Example: SMD Removal Using ChipQuik

The ChipQuik SMD Removal Kit (www.chipquik.com) allows you to quickly and easily remove surface mount components such as PLCC, SOIC, TSOP, QFP, and discrete packages. The primary component of the kit is a low-melting temperature solder (requiring less than 300 degrees F) that reduces the overall melting temperature of the solder on the SMD pads. Essentially, this enables you to just lift the part right off the PCB.

WARNING: HARDWARE HARM



Please read through this example completely before attempting SMD removal on an actual device. When removing the device, be careful to not scratch or damage any of the surrounding components or pull up any PCB traces. After following the instructions on the package (which consists of simply applying a standard no-clean flux to the SMD pins and then applying a low-melting-point solder), you can easily remove the surface mount part from the board.

Figure A.33 shows the contents of the basic ChipQuik SMD Removal Kit, from top to bottom: alcohol pads for cleaning the circuit board after device removal, the special low-melting temperature alloy, standard no-clean flux, and application syringe.

Figure A.33 ChipQuik SMD Removal Kit Contents



Figure A.34 Circuit Board Before Part Removal

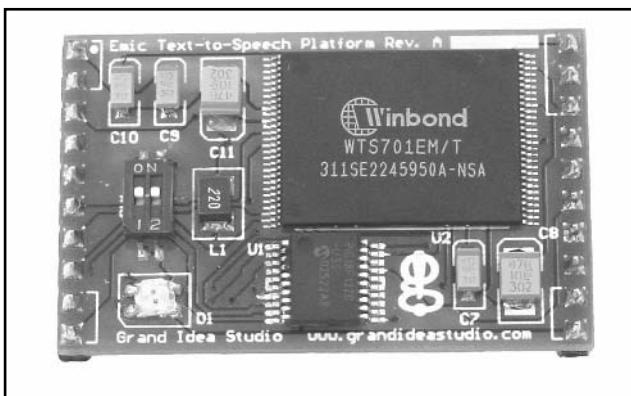


Figure A.35 Applying Flux to the Leads

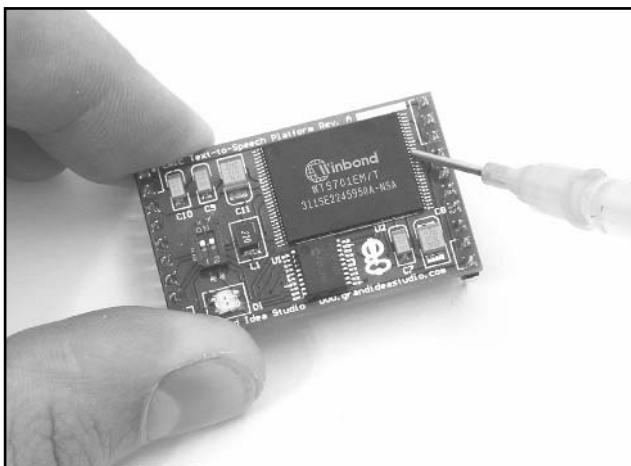


Figure A.36 Chip with Flux Applied

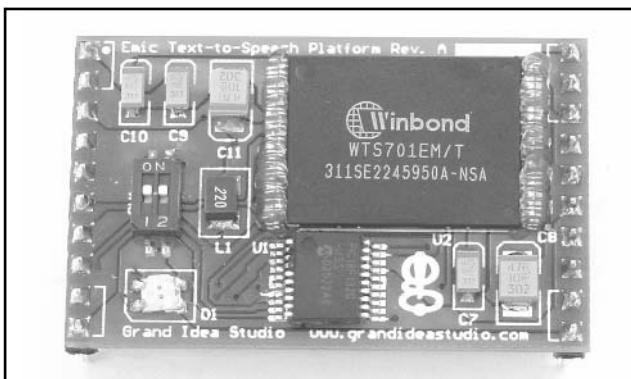


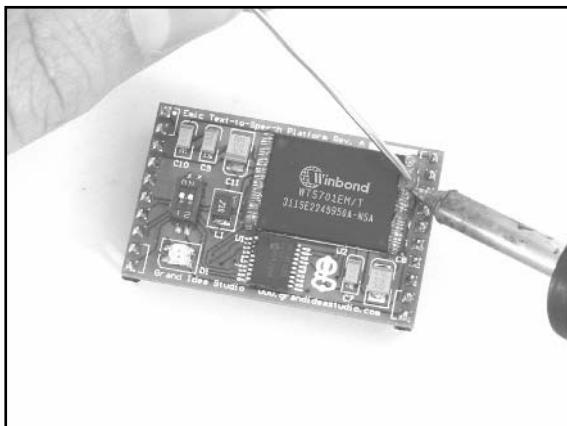
Figure A.34 shows the circuit board before the SMD part removal. Our target device to remove is the largest device on the board, the Winbond WTS701EM/T 56-pin TSOP IC.

The first step is to assemble the syringe, which contains the no-clean flux. Simply insert the plunger into the syringe and push down to dispense the compound (see Figure A.35). The flux should be applied evenly across all the pins on the package you will be removing (see Figure A.36). Flux is a chemical compound used to assist in the soldering or removal of electronic components or other metals. It has three primary functions:

1. Cleans metals surfaces to assist the flow of filler metals (solder) over base metals (device pins)
2. Assists with heat transfer from heat source (soldering iron) to metal surface (device pins)
3. Helps in the removal of surface metal oxides (created by oxygen in the air when the metal reaches high temperatures)

Once the flux is evenly spread over the pins of the target device, the next step is to apply the special ChipQuik alloy to the device (see Figure A.37). This step is just like soldering: Apply heat to the pins of the device and the alloy at the same time. The alloy has a melting point of approximately 300 degrees F, which is quite low. You should not have to heat the alloy with the soldering iron for very long before it begins to melt. The molten alloy should flow around and under the device pins (see Figure A.38).

Figure A.37 Applying Heat and Alloy to the Leads



Now that the alloy has been properly applied to all pins of the device, it is time to remove the device from the board. After making sure that the alloy is still molten by reheating all of it with the soldering iron, gently slide the component off the board (see Figure A.39). You can use a small jeweler's flathead screwdriver to help with the task. If the device is stuck, reheat the alloy and wiggle the part back and forth to help the alloy flow underneath the pads of the device and loosen the connections.

The final step in the desoldering process is to clean the circuit board. This step is important because it will remove any impurities left behind from the ChipQuik process and leave you ready for the next step in your hardware hack.

First, use the soldering iron to remove any stray alloy left on the device pads or anywhere else on the circuit board. Next, apply a thin, even layer of flux to all of the pads that the device was just soldered to. Use the included alcohol swab or a flux remover spray to remove the flux and clean the area (see Figure A.40).

Starting at one end of the device, simply heat and apply the alloy. Repeat for the other side(s) of the device. The flux will help with ensuring a nice flow of the alloy onto the device pins. Ensure that the alloy has come in contact with every single pin by gently moving the soldering iron around the edges of the device. Avoid touching nearby components on the PCB with the soldering iron.

Figure A.38 Chip with Alloy Applied

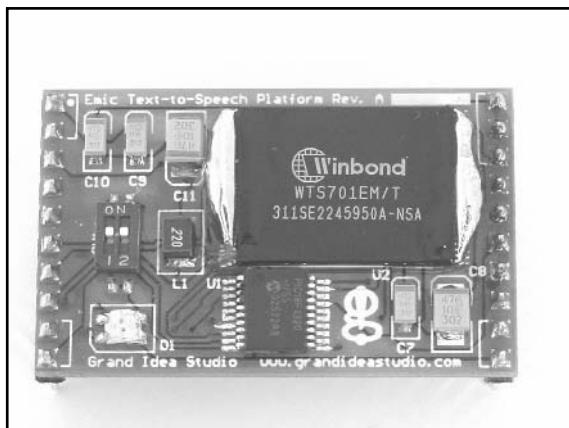


Figure A.39 Removing the Device from the Board

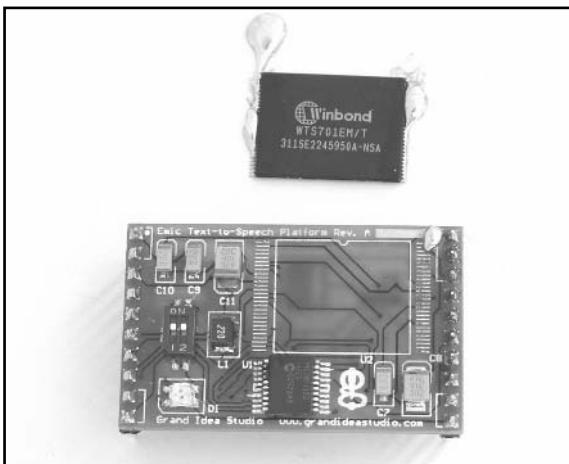


Figure A.40 Using Flux and Alcohol Swab to Clean Area

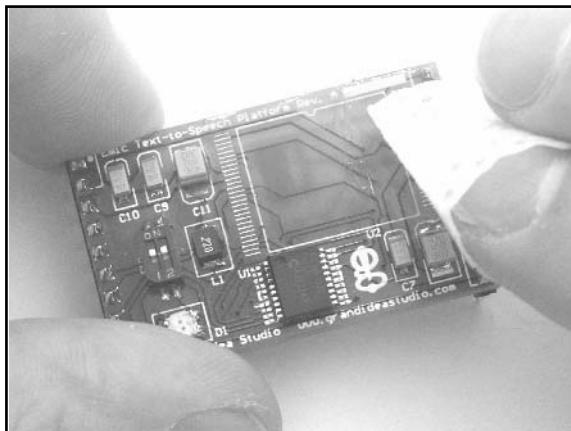
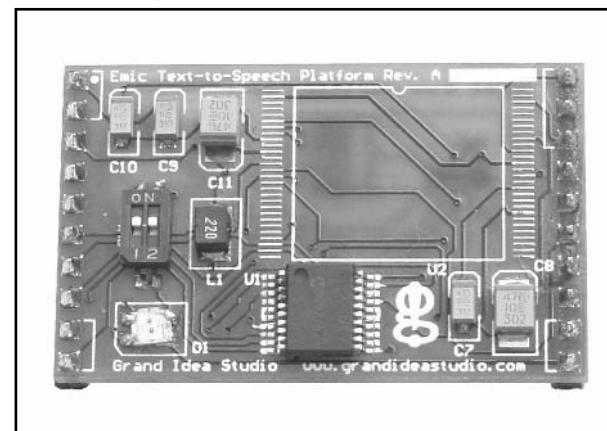


Figure A.41 Circuit Board with Part Successfully Removed



The desoldering process is now complete. The surface mount device has been removed and the circuit board cleaned (see Figure A.41). If you intend to reuse the device you just removed, use the soldering iron to remove any stray alloy or solder left over on and in between the pins and ensure there are no solder bridges between pins. If you do not want to reuse the device, simply throw it away.

Common Engineering Mistakes

During engineering design and debugging, you should remember the important maxim K.I.S.S.—or Keep It Simple, Stupid—at all times. It can be frustrating to troubleshoot a problem for hours or days on end and then discover the cause was a simple oversight. The most common engineering mistakes for hardware hacking are listed here. Although there are hundreds of other simple mistakes that can cause an engineer to quickly lose his or her hair, this list should get you started:

- **Faulty solder connections** After soldering, inspect the connections for cold solder joints and solder bridges. Cold solder joints happen when you don't fully heat the connection or when metallic corrosion and oxide contaminate a component lead or pad. Cold solder joints are the most common mistake for amateur and hobbyist electronics builders. Solder bridges form when a trail of excess solder shorts pads or tracks together (see the “Soldering Techniques” section in this appendix).
- **Installing the wrong part** Verify the part type and value before you insert and solder the component to the circuit board. Although many devices appear to look similar (e.g., a 1K and a 10K resistor look almost the same except for the color of one band), they have different operating characteristics and may act very differently in an electronic circuit. Surface mount components are typically harder to distinguish from one another. Double check to ensure that each part is installed properly. Keep in mind that the only way to properly test a component’s value is to remove it from the board and then test it.

- **Installing parts backward** ICs have a notch or dot at one end indicating the correct direction of insertion. Electrolytic capacitors have a marking to denote the negative lead (on polarized surface mount capacitors, the positive lead has the marking). Through-hole capacitors also have a shorter-length negative lead than the positive lead. Transistors have a flat side or emitter tab to help you identify the correct mounting position and are often marked to identify each pin. Diodes have a banded end indicating the cathode side of the device.
- **Verify power** Ensure that the system is properly receiving the desired voltages from the power supply. If the device uses batteries, check to make sure that they have a full charge and are installed properly. If your device isn't receiving power, chances are it won't work.

Web Links and Other Resources

General Electrical Engineering Books

- Radio Shack offers a wide variety of electronic hobby and “how to” books, including an Engineer’s Notebook series of books that provide an introduction to formulas, tables, basic circuits, schematic symbols, integrated circuits, and optoelectronics (light-emitting diodes and light sensors). Other books cover topics on measurement tools, amateur radio, and computer projects.
- *Nuts & Volts* (www.nutsvolts.com) and *Circuit Cellar* (www.circuitcellar.com) magazines are geared toward both electronics hobbyists and professionals. Both are produced monthly and contain articles, tutorials, and advertisements for all facets of electronics and engineering.
- Horowitz and Hill, *The Art of Electronics*, Cambridge University Press, 1989, ISBN 0-52-137095-7. Essential reading for basic electronics theory. It is often used as a course textbook in university programs.
- C. R. Robertson, *Fundamental Electrical & Electronic Principles*, Newnes, 2001, ISBN 0-75-065145-8. Covers the essential principles that form the foundations for electrical and electronic engineering courses.
- M. M. Mano, *Digital Logic and Computer Design*, Prentice-Hall, 1979, ISBN 0-13-214510-3. Digital logic design techniques, binary systems, Boolean algebra and logic gates, simplification of Boolean functions, and digital computer system design methods.
- K. R. Fowler, *Electronic Instrument Design*, Oxford University Press, 1996, ISBN 0-19-508371-7. Provides a complete view of the product development life cycle. Offers practical design solutions, engineering trade-offs, and numerous case studies.

Electrical Engineering Web Sites

- **ePanorama.net:** www.epanorama.net A clearing house of electronics information found on the Web. The content and links are frequently updated. Copious amounts of information for electronics professionals, students, and hobbyists.
- **The EE Compendium, The Home of Electronic Engineering and Embedded Systems Programming:** <http://ee.cleversoul.com> Contains useful information for professional electronics engineers, students, and hobbyists. Features many papers, tutorials, projects, book recommendations, and more.
- **Discover Circuits:** www.discovercircuits.com A resource for engineers, hobbyists, inventors, and consultants, Discover Circuits is a collection of over 7,000 electronic circuits and schematics cross-references into more than 500 categories for finding quick solutions to electronic design problems.
- **WebEE, The Electrical Engineering Homepage:** www.web-ee.com Large reference site of schematics, tutorials, component information, forums, and links.
- **Electro Tech Online:** www.electro-tech-online.com A community of free electronic forums. Topics include general electronics, project design, microprocessors, robotics, and theory.
- **University of Washington EE Circuits Archive:**
www.ee.washington.edu/circuit_archive A large of collection of circuits, data sheets, and electronic-related software.

Data Sheets and Component Information

When reverse engineering a product for hardware hacking purposes, identifying components and device functionality is typically an important step. Understanding what the components do may provide detail of a particular area that could be hacked. Nearly all vendors post their component data sheets on the Web for public access, so simple searches will yield a decent amount of information. The following resources will also help you if the vendors don't:

- **Data Sheet Locator:** www.datasheetlocator.com A free electronic engineering tool that enables you to locate product data sheets from hundreds of electronic component manufacturers worldwide.
- **IC Master:** www.icmaster.com The industry's leading source of integrated circuit information, offering product specifications, complete contact information, and Web site links.
- **Integrated Circuit Identification (IC-ID):** www.elektronikforum.de/ic-id Lists of manufacturer logos, names, and datecode information to help identifying unknown integrated circuits.

- **PartMiner:** www.freetradezone.com Excellent resource for finding technical information and product availability and for purchasing electronic components.

Major Electronic Component and Parts Distributors

- Digi-Key, 1-800-344-4539, www.digikey.com
- Mouser Electronics, 1-800-346-6873, www.mouser.com
- Newark Electronics, 1-800-263-9275, www.newark.com
- Jameco, 1-800-831-4242, www.jameco.com

Obsolete and Hard-to-Find Component Distributors

When trying to locate obscure, hard-to-find materials and components, don't give up easily. Sometimes it will take hours of phone calls and Web searching to find exactly what you need. Many companies that offer component location services have a minimum order (upward of \$100 or \$250), which can easily turn a hobbyist project into one collecting dust on a shelf. Some parts-hunting tips:

- Go to the manufacturer Web site and look for any distributors or sales representatives. For larger organizations, you probably won't be able to buy directly from the manufacturer. Call your local distributor or representative to see if they have access to stock. They will often sample at small quantities or have a few-piece minimum order.
- Be creative with Google searches. Try the base part name, manufacturer, and combinations thereof.
- Look for cross-reference databases or second-source manufacturers. Many chips have compatible parts that can be used directly in place.

The following companies specialize in locating obsolete and hard-to-find components. Their service is typically not inexpensive, but as a last resort to find the exact device you need, these folks will most likely find one for you somewhere in the world:

- USBid, www.usbid.com
- Graveyard Electronics, 1-800-833-6276, www.graveyardelectronics.com
- Impact Components, 1-800-424-6854, www.impactcomponents.com
- Online Technology Exchange, 1-800-606-8459, www.onlinetechx.com

Index

Symbols

- * (asterisk), 561
- = (equals sign), 561
- #GP32 IRC Channel, 287
- #gp32dev IRC Channel, 283
- (minus sign), 561
- + (plus sign), 561
- / (slash character), 561
- \ (backslash character), 557
- \\" (double quotation mark), 558
- \? (question mark), 558
- \' (single quotation mark), 558
- \a (alert-bell character), 558
- \n (new-line character), 557–558
- \t (horizontal tab character), 558
- \{} (curly braces), 557
- \\tau (time constant, tau), 524
- \Omega (resistance symbol, omega), 519

Numerals

- 5-pin plastic header, 162
- 10 Yard Fight game, 324
- 10NES security mechanism, 311–312
- 802.11 Wireless Networks: *The Definitive Guide* (book), 123
- 802.11b adapter, adding to Xbox, 123–125
- 2600 101 Web site, 396
- 2600 Programming for Newbies Web site, 395–396
- 5200BAS Compiler Web site, 474

A

- AC (alternating current), 517–518
- Accelerating the GP32, 264–269
- accupc.com
 - motherboard, 29
 - Web site, 83
- Activision, 487, 489
- Ad hoc networks, 124
- Address operators, 568
- Adjustable power supplies, 13–14
- Administr8or Dashboard, 140

- Advanced Atari Gaming Studies, Institute for, 506
- Advanced RISC Machine (ARM) of GBA, 226–227
- Advanced tools, 13–16
- Afterburner GBA Internal Lighting Kit, 198–200
- Ah! Catsmanga Daioh game, 281
- Air, compressed, 30
- Airflow management, 20
- Alert-bell character (\a), 558
- Alloy, ChipQuik, 534–535
- Alternating current (AC), 517–518
- American Standard Code for Information Interchange (ASCII), 512–514
- Amperes (amps), 516–517
- Analog multimeters, 13
- Analyzers, logic, 16
- AnimalSNES Web site, 83
- Antenna, removable, for Xbox, 125–131
- Antenna jack, 131
- Antistatic precautions, 33, 34, 67, 153, 188, 251, 415
- Arcadia, 393
- Architecture, parallel, 171
- Arkanoid game, 324
- ARM (Advanced RISC Machine) of GBA, 226–227
- Arocha, Chris, 125
- Arrays, 549–550, 567–570
- The Art of Assembly Language* (book), 584
- The Art of Electronics* (book), 537
- ArTile Micro TX79 processor, 171, 173
- ASCII (American Standard Code for Information Interchange), 512–514
- Assembly language
 - data types, 581
 - description, 578
 - jumps, 580
 - labels, 579–580
 - operands, 581
 - operations, 580
 - sample program, 582–583
- stacks, 583–584
- strings, 577, 581
- Assignment, 543–544, 561
- Asterisk (*), 561
- Atari. *see* Atari 2600 (VCS); Atari 2600, converting to PC; Atari 2600 joystick hacks; Atari 5200 four-port switchbox; Atari 5200 paddle controller; Atari 5200 SuperSystem; Atari 5200 video and audio upgrade; Atari 7800 ProSystem; Atari CX-30 paddle controllers; Atari CX-40 joysticks; Atari History Museum
- Atari 800 ITX Web site, 83
- Atari 2600 (VCS)
 - audio/video modification, 380
 - case, cleaning, 31–32
 - case, closing, 78–81
 - case, opening, 29–30
 - CD-ROM drive, 34, 67–70
 - Commodore 1702 monitor, 380
 - compatibility with 7800 games, 478, 501–502
 - control panel, 75–78
 - FAQ, 502
 - hard drive, 36, 71–72
 - history, 336–337
 - homebrew game development, 391–396
 - models, 27
 - motherboard, 24–25, 28–29, 70–71
 - NES control pad, 356–363
 - paddle maintenance, 349–356
 - power supply, 496
 - PW70 Power Supply Module, 72–73
 - reassembly, 67–78, 379–380, 390
 - renamed, 21
 - s-video/audio modification, 364–380
 - solder pad pinout, 372
 - speakers, 382
 - stereo audio output, 382–391
 - USB components, 73–75

workshop Web site, 502, 505
see also Atari 2600, converting to PC; Atari 2600 joystick hacks
 Atari 2600, converting to PC
 ATX power connector, 56–59
 ATX Power Supply Module, iTuner PW70, 54–55, 72–73
 back panel, 23
 BIOS, configuring, 35–36
 case modifications, resources for, 82
 cleanup, preparation for, 24
 components, attaching, 33–34
 control panel, preparing, 38–45
 cordless keyboard/mouse receiver, 26, 46–50, 74, 77
 cost of hack, 23
 critique, post-hack, 82
 description, 21–22
 design mockup, 32
 DRAM slots, 66–67
 dust cover, 82
 fan, 29
 housing, 59–67
 modifications, laptop, 149
 parts and materials, 83
 portable game systems, resources for, 83
 power switch, 76
 powering on, 34–35
 preparation for hack, 23–29
 reassembly, 67–78
 resources, 82–83
 software, installing, 36–38
 Stelladaptor 2600 Controller-to-USB Interface, 22–24, 51–53, 74, 78
 switches, 22, 30
 tools, 26–27
 TV output, 37
 USB/FireWire backplane, preparing, 45–46
 Atari 2600 joystick hacks
 description, 337–338
 joystick modification, left-handed, 337–342
 joysticks, repairing, 342–349
 preparation for hacks, 338, 342
 tools, 338, 342

Atari 2600 Programming Resources Web site, 396
 Atari 5200 four-port switchbox
 capacitor, 443
 description, 434–435
 explanation of hack, 445–446
 power cable, 440–442
 preparation for hack, 435
 removing components, 436–438
 RF cable, 444
 tools, 436
 wiring, 438–440
see also Atari 5200 SuperSystem
 Atari 5200 paddle controller
 building Atari 5200 paddle controller, 424–431
 description, 419–421
 disassembling Atari 2600 paddle controller, 422–423
 explanation of hack, 432–433
 weighted dial, 432–433
see also Atari 5200 SuperSystem
 Atari 5200 SuperSystem
 blue LED modification, 408–413
 case, opening, 401–408
 controllers, rebuilding, 467–470
 explanation of LED hack, 413
 four-port VCS cartridge adapter fix, 470
 game development, homebrew, 470–474
 history, 400, 478
 other hacks, 467–470
 preparations for hacks, 401, 409
 reassembly, 408
 tools, 410
 two-port BIOS replacement, 413–419
see also Atari 5200 four-port switchbox; Atari 5200 paddle controller; Atari 5200 video and audio upgrade
 Atari 5200 video and audio upgrade
 access, 449–451
 description, 446–447
 GTIA (George's Television Interface Adapter), 459–461
 jacks, 451–456

POKEY chip, 463–465
 preparation for hack, 447–448
 tools, 448
 Video Upgrade Board, 461–463
 wiring, 456–458
 Atari 7800 ProSystem
 Atari7800.org Web site, 506
 BIOS replacement, 534
 blue LED modification, 479–485
 case, opening, 481
 compatibility with 2600 games, 478, 501–502
 composite and s-video output, 501
 DevOS modification and cable creation, 502
 explanations of hacks, 485–486, 489, 494–495
 game compatibility hacks, 486–490
 High Score Cart, 504
 history, 478–479
 homebrew game development, 502–506
 NES control pad modification, 502
 power supply plug retrofit, 495–501
 preparations for hacks, 480, 487, 490, 496
 reassembly, 484, 489, 494, 500
 Sega Genesis controller modification, 501–502
 Signature Key Generator, 505
 Tech Page Web site, 505
 tools, 480, 487, 491, 497
 validation key, 502–503
 voltage regulator replacement, 490–495
 Web resources, 506–507
 Atari CX-30 paddle controllers, 349
 Atari CX-40 joysticks
 common failures, 342
 description, 337–338
 reassembly, 341, 347–348
 replacement parts, 342–343
 wire colors, 340, 347

Atari History Museum, 396–397, 475, 504, 506
 Atari ST utility, 503
 Atari Times Web site, 397
 Atari800 Web site, 474
 Atari800MacX Web site, 474
 Atari800Win PLus Web site, 474
 AtariArchives.org Web site, 474
 AtariAge Web site, 394–396, 473–474, 506
 AtariMagazines.com Web site, 474
 AtariProtos.com Web site, 397, 475, 506–507
 Attributes, file, 591
 ATX power connector for Atari 2600, 56–59
 ATX Power Extension Cable, 56–59, 68–69
 ATX Power Supply Module, 54–55
 Audio/video modification of Atari 2600, 380
 Authentication, bypassing, 148
 AwardBIOS CMOS Setup Utility, 35

B

Back panel for Atari 2600, 23
 Backing up firmware, 276–277
 Backlight Unit (BLU) of GP32, 244
 Backslash (\), 557
 BADATA-SYSTEM folder, 166, 169
 Ball Grid Array (BGA), 528
 Ballblazer game, 479
 Bank switching, 330, 332, 486–487
 Bardeen, John, 526
 Base input current (I_B), 527
 Base voltage (V_B), 526
 BASIC, Family, 292
 Basic device theory, 519–529
 Basic Input/Output System (BIOS)
 configuring for Atari 2600, 35–36
 replacing for Atari 5200, 413–419
 replacing for Atari 7800, 534
 Batteries
 designators and symbols, 515
 heating, 366
 replacing, 319–323

replacing in game cartridges, 319–323
 B&C ComputerVisions Web site, 397, 475, 507
 BEDATA-SYSTEM folder, 166, 169
 Berkeley, University of California at, 597
 Bernina Artista 200E sewing machine, 600
 Best Electronics, 82, 83, 342, 397, 475, 507
 BGA (Ball Grid Array), 528
 BIDATA-SYSTEM folder, 166, 169
 Bidots!, 28
 Bigass NES Mapper List, 330
 BIOS. *see* Basic Input/Output System (BIOS)
 Bitmap-based modes of GBA, 232
 Bits
 drill, 7
 NES 3.8mm security bit (gamebit), 316, 318
 security driver, 9
 Bits, bytes, and nibbles, 510–514
 Bits (memory), 588
 Block devices, 595–596
 BLU (Backlight Unit) of GP32, 244
 Blue LED modification
 Atari 5200, 408–413
 Atari 7800, 479–485
 NES 2, 302–311
 Blue screen flashing, NES 2, 299, 302, 315
A Book on C, 584
 Books
 802.11 Wireless Networks: The Definitive Guide, 123
 The Art of Assembly Language, 584
 The Art of Electronics, 537
 A Book on C, 584
 C Programming Language, 584
 Designing a Wireless Network, 123
 Digital Logic and Computer Design, 537
 electrical engineering, 537
 Electronic Instrument Design, 537
 Fundamental Electrical & Electronic Principles, 537
 Game Over: Press Start to Continue, 292
 Getting Started with Windows Drivers, 596
 Hack Proofing Your Wireless Network, 123
 Hacking the Xbox, 144
 Installing Troubleshooting and Repairing Wireless Networks, 123
Jeff Duntemann's WiFi Guide, 123
Linux Device Drivers, 596
Modern Operating Systems, 594, 601
Programming from the Ground Up, 584
 Radio Shack, 537
Structured Computer Organization, 584
Ultimate Wireless Reference CD, 123
 Booleans, 559
 BOOT.ELF file, 166, 168
 Braces, curly ({}), 557
 Branching
 conditional, 545–546
 unconditional, 546–548
 Brattain, Walter, 526
 Bratwurst Toaster Web site, 332
 BSD software, 597
 Buttons, GP32, repairing, 262–264
 Bypassing hardware authentication, 148
 Bytes, 510–514
 Bytes (memory), 588

C

C language
 comparison operators, 564
 compilers, 556
 control structures, 562–567
 data types, 558–559
 description, 555
 for loops, 562–563
 function calls, 573–574
 hardware access, 574
 history, 556
 if-else statements, 565–566
 increment/decrement operators, 564

- mathematical functions, 559–562
 operators, other, 575
 printing to the screen, 556–558
 storage structures, 567–573
switch statements, 566–567
 system calls, 574
 variables, passing, 573–574
while loops, 564–565
- C Programming Language* (book), 584
- Cable management, 20
- Cables
 ATX power, 56–59, 68–69
 joysticks, 24, 53
 serial input/output (SIO), 157
- Cache memory, 591–592
- Call-by-value passing, 574
- Camerica, 312
- Capacitors
 description, 521–524
 designators and symbols, 514–515
- Carnegie Mellon University, 331
- Cartridge connector, 72-pin, 299–302
- Cartridge-loading mechanism, NES 2, 297–298
- Cartridges, game. *see* Game cartridges; Games; Nintendo Entertainment System (NES)
 game cartridges
- Case Mod Supplies Web site, 83
- Case windows, 20
- Cases
 cleaning, 31–32
 lighting for, 20
 modifications, laptop, 149
 modifications, resources for, 82
 opening Atari 2600, 29–30
 opening Atari 5200 SuperSystem, 401–408
 opening Atari 7800 ProSystem, 481
 opening Game Boy Advance (GBA), 187–193
 opening GamePark 32 (GP32), 251–257
 opening NES game cartridges, 316–319
- opening PlayStation 2 (PS2), 152–156
 opening Xbox, 92–95
 silencing, 20
- Castle Crisis game, 420, 473
- CD-ROM drive, Atari 2600, 34, 60–61, 67–70
- CDs, playing, 22
- Center punches, 10
- Central processors (CPUs)
 core voltage increase (accelerating), 264–269
 GBA, 226–227
 GBA registers, 227–228
 sub-CPU Interface (SIF), 176
- Char data type, 559
- Character devices, 594–595
- Character strings, 559
 in assembly language, 577, 581
 in C, 571–572
- Charge, storing, 521
- Chip, lockout. *see* Lockout Chip, NES
- Chip-on-Board (COB) packaging, 529
- ChipQuik SMD Removal Kit, 532–534
- CHR-ROM image, 328, 330
- Chrominance (chroma), 364, 382
- Circuit Cellar* magazine, 537
- Cleaning contacts, 302
- Cleaning supplies, 7
- Cleanup, preparation for, 24
- CLI (command-line interface), 593
- Climber 5 game, 392
- Clippers, wire, 10
- Clock speed, 264
- Coats, lab, 5
- COB (Chip-on-Board) packaging, 529
- Cobbleware, 264
- Codemasters, 312
- Cold solder joints, 531, 536
- Collector-emitter current (I_C), 527
- Collector voltage (V_C), 526
- Color-coding
 resistors, 519–520
 wires, joystick, 340, 347
- Command-line interface (CLI), 593
- Commands, preprocessor, 577–578
- Comments, 553–554, 560
- Commodore 1702 monitor, 380
- Common mistakes, 536–537
- Comparison operators, 564
- Component video standards, 381
- Components
 distributors, 539
 electronic, 12
 information about, 538–539
- Composite audio/video output
 Atari 7800 ProSystem, 501
 Nintendo Entertainment System (NES 2 or Top Loader), 332
- Composite Audio/Video Output for NES 2 Web site, 332
- Composite video standards, 382
- Comprehensive NES Mapper Document, 330
- Compressed air, 30
- Conditional branching, 545–546
- Configuration
 BIOS for Atari 2600, 35–36
 GP32, 245–249
 motherboard BIOS, Atari 2600, 35–37
- Connector shield, 62
- Connectors
 adapters, 502
 cartridge (72-pin), 299–302
 odd, 495
 PS/2, 49
 pulling, 340, 346
see also Power connectors
- Constants, symbolic, 562
- Contact cleaner, 353
- Contact East, 17
- Contacts, cleaning, 302
- Control pad, NES. *see* Nintendo Entertainment System (NES) control pad
- Control panel, Atari 2600, 38–45, 75–78
- Control structures, 544–548, 562–567
- Controller-to-USB interface. *see* Stelladaptor 2600 Controller-to-USB Interface
- Controllers, game

Atari CX-30 paddle, 349
 Sega Genesis modification, 501–502
 Xbox, 95–104
 Conversion to PC, Atari 2600
 description, 21–22
 Web resources, 83
 Cordless drills, 7
 Cordless keyboard/mouse receiver, 26, 46–50, 74, 77
 Core fundamentals of electronics, 510–519
 Core voltage increase (accelerating), 264–269
 Coulombs, 516–517
 Coupler, s-video, 365
 Cover, Atari, 82
 CPU. *see* Central processors (CPUs)
 Crazy Jack game, 281
 Critique, post-hack
 Atari 2600 (VCS), 82
 Crossover cable, making, 116–120
 Crystal designators and symbols, 515
 Curly braces ({}), 557
 Current, electrical
 description, 516–518
 forward (I_F), 525
 Current-limiting resistor, 310–311, 485–486, 516, 520, 526
 Curve, diode V-I, 525
 Custom skin for GBA, 237–238
 Cuttle Cart cartridge tool, 393, 489, 496, 504–506
 CX-30 paddle controllers, 349
 CX-40 joysticks. *see* Atari CX-40
 joysticks
 Cxbx, The Xbox Emulator Web site, 145
 CyberTech, 364

D

Dan B's Atari 5200 Tech Page, 473
 Dan Peori's Web site, 176
 Dark Chambers game, 486, 489–490
 Dark Fader's DumpFW, 276

Dark Fader's Gamepark 32 Web site, 283
 DASM Assembler Web site, 396
 Data, displaying in C, 556–558
 Data sheets, 538–539
 Data types
 in assembly language, 581
 in C, 558–559
 Davie, Andrew, 395–396
 Day of the Tentacle game, 281
 DC. *see* Direct current (DC)
 DC power adapter, creating, 269–275
 DDD debugger front-end, 576
 Debugger, GDB, 156
 Debugging
 printf command, 576–578
 tools, 575–576
 Decathlon game, 489
 Decrement/increment operators, 564
 Design mockup, Atari 2600 hack, 32
Designing a Wireless Network (book), 123
 Desk lamps, 5
 Desoldering, 532–536
 Desoldering switches, 38–39
 Development of games. *see* Game development, homebrew
 Development Operating System (DevOS)
 modification and cable creation for Atari 7800, 502
 Device drivers, 593–594
 Device programmers, 14–15
 DevKitAdvance software development kit, 281
 DevOS. *see* Development Operating System (DevOS)
 Dial, weighted, 432–433
 Dielectric, 521–522
 The Dig game, 281
 Digi-Key Corporation
 components, 17, 539
 Web site, 83
Digital Logic and Computer Design (book), 537
 Digital multimeters, 13
 Dijkstra, Edsger W., 592
 Dining philosophers problem, 592
 Diodes
 description, 524–526
 designators and symbols, 515
 V-I curve, 525
 Direct current (DC)
 blocking, 521
 description, 517–518
 Direct memory access (DMA), 169, 173, 176
 Directron.com Web site, 83
 Discover Circuits Web site, 538
 Display lens, replacing in GBA, 193–198
 Display setting, Windows, 37
 Displaying data in C, 556–558
 Distributors, 539
 DMA (direct memory access), 169, 173, 176
 Dodgson, Harry, 503
 Donkey Kong and Donkey Kong Jr. games, 324
 DOOM game, 281
 Doping, 526
 Double data type, 559
 Double quotation mark (\"), 558
 Dragon Warrior I and II games, 319
 DRAM slots in Atari 2600, 66–67
 Dreamcast PC Web site, 83
 Dremel tools, 6
 Drills and bits, 7
 Duck Hunt game, 324
 Duct tape, 7
 Dust cover, Atari, 82
 DVD drive, Atari 2600, 60–61
 DVDs, playing, 22, 82

E

e-Reader, 235–237
 Earth (ground) designators and symbols, 516
 eBay, 28
 EE Circuits Archive Web site, 538
 EE Compendium Web site, 538
 EE (Emotion Engine), PS2, 156, 164, 171–175

802.11 *Wireless Networks: The Definitive Guide* (book), 123
 802.11b adapter, adding to Xbox, 123–125
 Electrical tape, 7
 Electro Tech Online Web site, 538
 Electronic components, 12
Electronic Instrument Design (book), 537
 Electronics, fundamentals of, 510–519
 Electrostatic discharge (ESD) protection. *see* Antistatic precautions
Embedded Linux Journal Web site, 601
 Embedded systems, 598–599
 Embedded.com, 601
 Emitter voltage (V_E), 526
 Emotion Engine (EE), PS2, 156, 164, 171–175
 ePanorama Web site, 538
 EPIA Nehemiah M10000 motherboard, 24–25, 28–29
 EPROM. *see* Erasable Programmable Read-Only Memory (EPROM)
 Equals sign (=), 561
 Erasable Programmable Read-Only Memory (EPROM)
 cartridge, creating, 324–330, 332
 development cartridge, 329–332
 UV-erasable, 15, 324
 Escape sequences, 558
 ESD (electrostatic discharge). *see* Antistatic precautions
 Essential tools, 5–8
 Etching kits for printed-circuit boards, 15
 European GP32 Site, 287
 Evolution X Administr8or Dashboard, 140
 EWRAM (External Working RAM) of GBA, 229–230
 Excitebike game, 324
 Explanations of hacks accelerating GP32, 268–269
 Atari 2600 audio/video modification, 391

Atari 5200 blue LED modification, 413
 Atari 5200 four-port switchbox, 445–446
 Atari 5200 paddle controller, 432–433
 Atari 7800 ProSystem, 485–486, 489, 494–495
 DC power adapter, creating, 275
 Game Boy Advance (GBA) internal lighting, 216
 PlayStation 2 (PS2), booting from memory card, 169–170
 PlayStation 2 (PS2), installing serial port in, 164
 removable antenna, adding to Xbox, 131
 Xbox controller, illuminating buttons, 99
 Exploit Installer, 165, 168
 Extension cables ATX power, 56–59, 68–69
 joysticks, 24, 53
 Extensions, file, GP32 and PC-related, 282
 External Working RAM (EWRAM) of GBA, 229–230

F

F8 bank-switching scheme, 486
 Famicom Console, Nintendo. *see* Nintendo Famicom Console
 Family BASIC programming language, 292
 Fan, 28–29
 Farads, 522
 File attributes, 591
 File extensions, GP32 and PC-related, 282
 File systems, 590–591
 Files, needle, 6–7
 Final Fantasy game, 319
find_title_params() routine, 170
 Firehawk game, 312
 FireWire Depot Web site, 83
 Firmware backing up, 276–277
 flashing, 278–280

5-pin plastic header, 162
 Flash memory devices, 590
 Flashing blue screen, NES 2, 299, 302, 315
 Flushing firmware, 278–280
 Fliptop case modification, 149
 Float data type, 559
 FLU (Frontlight Unit) of GP32, 244
 Flux, 534
 Foam tape, 71, 79
for loops, 562–563
 Form factor, Mini-ITX, 28
 Formatting code, 555
 Forward current (I_F), 525
 Frontlight Unit (FLU) of GP32, 244, 284
 fr★shgloop Web site, 145
 Function and variable names, 554
 Functions in C mathematical, 559–562
 programming, 573–574
Fundamental Electrical & Electronic Principles (book), 537
 Fundamentals of electronics, 510–519
 Funding for Atari 2600 (VCS) hack, 23
 Fuse designators and symbols, 515

G

Game boxes. *see* Atari; GamePark 32 (GP32); Nintendo; PlayStation 2 (PS2); Xbox (XBC)
 Game Boy Advance (GBA)
 Advanced RISC Machine (ARM), 226–227
 architecture, 184
 bitmap-based modes, 232
 case, opening, 187–193
 central processor (CPU) registers, 227–228
 custom skin, 237–238
 display lens, replacing, 193–198
 e-Reader codes, creating, 235–237

- External Working RAM (EWRAM), 229–230
 game development, homebrew, 233–234
 game ROM, 231
 game save memory, 231
 graphics system, 231
 hardware development tools, 233–234
 hardware specifications table, 182
 history, 182
 Internal Working RAM (IWRAM), 229–230
 loading UNIX, 235
 memory architecture, 228–229
 multiboot cable, 233–234
 Object Attribute Memory (OAM), 231
 other hacks, 234–238
 palette memory, 230
 power LED modification, 234–235
 reassembly, 193
 sound system, 231
 tile-based modes, 232
 video RAM (VRAM), 230
 Web resources, 238–239
see also Game Boy Advance (GBA) internal lighting; Game Boy models; Nintendo corporate history; Stealth Dimmer Chip
- Game Boy Advance (GBA) internal lighting
 Afterburner kit, 198–200
 Afterburner module, installing, 211–214
 Afterburner module, preparing, 209–210
 brightness control, 214–216
 explanations of hacks, 216, 225–226
 housing, preparing, 203–205
 LCD screen, preparing, 206–209
 LCD screen, removing, 201–203
see also Game Boy Advance (GBA); Stealth Dimmer Chip
- Game Boy models
- Advance SP, 185
 Color, 183–184
 Pocket, 183
see also Game Boy Advance (GBA); Nintendo corporate history
- Game cartridges
 Mapper 0 style (NROM), 324–325, 330
 NES 3.8mm security bit (gamebit), 316, 318
 size limitation, 486
see also Games; Nintendo Entertainment System (NES) game cartridges
- Game development, homebrew
 Atari 2600, 391–396
 Atari 5200 SuperSystem, 470–474
 Atari 7800, 502–506
 Game Boy Advance (GBA), 233–234
 GamePark 32 (GP32), 280–283
 Nintendo Entertainment System game cartridges, 324–332
 PlayStation 2 (PS2), 176
 university course, 331
 Web sites, 176, 331
 Xbox, 144–145
- Game Over: Press Start to Continue* (book), 292
- Game ROM of GBA, 231
 Game save memory of GBA, 231
 Gamebit (NES 3.8mm security bit), 316, 318
- GamePark 32 (GP32)
 accelerating, 264–269
 Backlight Unit (BLU), 244
 buttons, repairing, 262–264
 case, opening, 251–257
 configuration, 245–249
 DC power adapter, creating, 269–275
 description, 243
 explanations of hacks, 268–269
 file extensions, 282
 firmware, flashing, 278–280
 firmware backup, 276–277
 first experience, 250
- Frontlight Unit (FLU), 244, 284
 game development, homebrew, 280–283
 Multifirmware loader, installing, 275–280
 other hacks, 284–286
 reassembly, 257
 registration process, 245
 screen cover, replacing, 257–262
 SmartMedia external memory card, 243
 test mode, 284–286
 Web resources, 286–287
- Gamepark Holdings Web site, 287
 Gamepark32.co.uk Web site, 287
- Games
 10 Yard Fight, 324
 Ah! Catsmanga Daioh, 281
 Arkanoid, 324
 Atari 2600/7800 compatibility, 478, 501–502
 Ballblazer, 479
 Castle Crisis, 420, 473
 Climber 5, 392
 compatibility hacks, 486–490
 Crazy Jack, 281
 Dark Chambers, 486, 489–490
 Day of the Tentacle, 281
 Decathlon, 489
 The Dig, 281
 Donkey Kong and Donkey Kong Jr., 324
 DOOM, 281
 Dragon Warrior I and II, 319
 Duck Hunt, 324
 Excitebike, 324
 Final Fantasy, 319
 Firehawk, 312
 Gorf, 419
 homebrew, 392
 Ice Hockey, 324
 Indiana Jones, 281
 K-Razy Shootout, 413
 Kaboom!, 419
 The Legend of Zelda, 319
 Marble Craze, 392
 Monkey Island, 281
 Moon Patrol, 419

- Mountain King, 413
 Pitfall, 413
 Pitfall II, 391
 Quattro Adventure, 312
 Robot Tank, 489
 Skeleton+, 391
 Slalom, 324
 Space Shuttle, 489
 StarFire, 394–395
 Super Breakout, 419
 Super Mario Brothers, 292, 324
 Supercharger, 489
 Tecmo Super Bowl, 319
 Thrust+ Platinum, 392
 title IDs for PS1 games, 166–167
 Ultima: Exodus, 319
 Zelda II: Link’s Adventure, 319
see also Game development,
 homebrew; Nintendo
 Entertainment System (NES)
 game cartridges
- GameSpy GBA Forums Web site, 238
 GB (gigabytes), 588
 GBA. *see* Game Boy Advance (GBA)
 GBAX.com Web site, 239, 287
 gdb command-line debugger, 575–576
 GDB debugger, 156
 GeePee32 Emulator Web site, 283
 George’s Television Interface Adapter (GTIA), 459–461
Getting Started with Windows Drivers (book), 596
 Gigabytes (GB), 588
 Glues, 7
 GNU Project, 597–598
 Goggles, 5
 Gorf game, 419
 GOTO statements, 579
 GP32. *see* GamePark 32 (GP32)
 GP32 add-on package, 282
 GP32 Devr’s Web site, 283
 GP32 FLU Dimmer Chip Hack Web site, 284
 GP32 GCC Toolchain in Linux Web site, 283
 #GP32 IRC Channel, 287
 GP32 JTAG Adapter Web site, 284
 GP32 World Web site, 287
- #gp32dev IRC Channel, 283
 gp32dev Yahoo! Group Web site, 283
 GP32Emu Web site, 283
 GP32Linux gP32oPie pre-alpha Web site, 284
 GP32Linux HOWTO Web site, 284
 GP32Linux Web site, 284
 GPAAdvance Web site, 283
 gpQuake Web site, 283
 Graphical user interfaces (GUIs), 593
 Graphics system of GBA, 231
 Graveyard Electronics, 539
 Graybill, Mark, 501
 Ground (earth) designators and symbols, 516
 GTIA (George’s Television Interface Adapter), 459–461
 GUIs (graphical user interfaces), 593
 Gun, Zapper, 292
- H**
- Hack Proofing Your Wireless Network* (book), 123
Hacking the Xbox (book), 144
 Hard disk drives (HDD)
 2.5-inch *versus* 3.5-inch, 29
 adding to Xbox, 144
 Atari 2600, 71–72
 partitioning, 36
 support cylinder, 59
 support software, 170–171
 Hard-to-find components, finding, 539
 Hardware access in C, 574
 Hardware authentication, bypassing, 148
 Hardware development tools, GBA, 233–234
 Hardware harm warnings
 batteries, heating, 366
 capacitor, removing, 437
 connectors, pulling, 340, 346
 contact cleaner, 353
 heat-sensitive components, 158
 holes, spacing, 387
 LCD screen, damaging, 208, 258
- lens, scratching, 194
 Mylar sheet, traces on, 425
 plastic posts, 404
 RF shield, bending, 406
 screw lengths, 402
 solder pads, 266, 306, 308, 325, 361, 410, 483
 stress on the board, 56
 traces, cutting, 406
 vias sinking, 160
 wire locations, 499
 wires, shielding, 440
see also Antistatic precautions
- Hardware stores, 17
 Harm to hardware, warnings of. *see* Hardware harm warnings
 Hash tables, 551
 HDD (hard disk drive) support, 170–171
 Head to Head Gaming, 125
 Header, 5-pin plastic, 162
 Heat guns, 9
 Heat-shrink tubing, 9
 Heat sinks
 grease (compound), 490, 493
 illustration, 491, 493
 reason for, 495
 wobbling on Atari 7800s, 490
 Hexadecimal (hex) format, 511–512
 Hierarchical file system, 590
 High Score Cart, Atari 7800, 504
 History
 Atari 7800 ProSystem, 478–479
 C language, 556
 Linux, 597–598
 Nintendo Entertainment Systems, 292
 UNIX, 597
 Hitachi Wearable Internet Appliance (WIA), 600
 Hobby Lobby, 17
 Home Depot, The, 17
 Homebrew game development. *see* Game development, homebrew
 Homebrew games
 Castle Crisis, 420, 473
 Climber 5, Marble Craze, and Thrust+ Platinum, 392

see also Game development,
homebrew
Horizontal tab character (\t), 558
Horton, Kevin, 330
Housing for Atari 2600, 59–67

I

I_B (base input current), 527
 I_C (collector-emitter current), 527
IC-ID (Integrated Circuit
Identification) Web site, 539
IC Master Web site, 538
Ice Hockey game, 324
IEC marking, 522–523
if-else statements, 565–566
 I_F (forward current), 525
IGN Game Boy Web site, 239
Impact Components, 539
Impedance, input, 13
Imperative languages, 542
Increment/decrement operators,
564
Independence exploit, PS2
description, 164, 170
reason for, 149–150
Web site, 165
Indiana Jones game, 281
Infrastructure networks, 124
Input impedance, 13
Input/output (I/O), 592
*Installing Troubleshooting and
Repairing Wireless Networks*
(book), 123
Institute for Advanced Atari
Gaming Studies, 506
Int data type, 559
Integrated Circuit Identification
(IC-ID) Web site, 539
integrated circuits (ICs), 528–529
Interface, controller-to-USB. *see*
Stelladaptor 2600
Controller-to-USB Interface
Internal Working RAM (IWRAM)
of GBA, 229–230
Interruptibility, 598
I/O processor (IOP), PS2, 175–176
Israel, Kirk, 396
ITuner PW70 ATX Power Supply
Module, 54–55, 72–73

IWRAM (Internal Working RAM)
of GBA, 229–230

J

Jack, antenna, 131
Jacks, power. *see* Power connectors
Jameco, 539
Jeff Duntemann's WiFi Guide (book),
123
Jigsaws, 10
Jnes NES Emulator Web site, 331
JoyGP Web site, 287
Joysticks
Atari 2600-compatible, 51
cables, 24, 53
color-coding wires, 340, 347
connectors, 30, 52
extension cables, 24, 53
left-handed modification,
337–342
see also Atari 2600 joystick hacks;
Atari CX-40 joysticks
Jum52 Web site, 474
Jumps in assembly language, 580
JunKmachine Web site, 333

K

K-Razy Shootout game, 413
Kaboom! game, 419
KB (kilobytes), 588
Key, validation, 502–503
Knee (of a diode), 525–526
Knives, X -ACTO, 6

L

Lab coats, 5
Labels in assembly language,
579–580
Lamps, 5
Language, strongly typed, 558
Languages, 542
LED. *see* Light-emitting diodes
(LEDs)
LED modification
Atari 5200, 408–413
Atari 7800, 479–485
Game Boy Advance (GBA),
234–235

NES 2, 302–311
Left-handed joystick modification.
see Atari 2600 joystick hacks

The Legend of Zelda game, 319
Lens, replacing in GBA, 193–198

Level shifter, 164
libHDD, 171

Light-emitting diodes (LEDs)
Atari 7800 ProSystem, 479–485
brightness, limiting, 485–486
designators and symbols, 515
example, 516
Game Boy Advance (GBA),
234–235
NES 1 (Toaster), 302–310
NES 2 (Top Loader), 302
Xbox, 120–123

Lighting
for cases, 20
types of, 5

Lik Sang Web site, 287
Linear voltage regulators, 495
Linked lists, 551–553
Linksys WET11 wireless bridge,
124

Linux
embedded (uCLinux), 598
file system, 590
GP32 GCC Toolchain, 283
history of, 597–598
installing in Xbox, 141–144
loading onto GP32, 284
open source, 597
PlayStation 2 (PS2) Linux Kit,
170, 173, 177

Linux Device Drivers (book), 596
Linux Devices Web site, 601
LisaZero PAL-only modchip, 149
Lists, linked, 551–553

Live Communicator, Xbox
adding to wireless controller,
111–112
reassembly, 110
remote reset switch, adding,
107–110

Loading Linux onto the GP32 Web
site, 284

Lockout Chip, NES
absence from NES 2, 312
disabling, 311–316

patents, 312
 toggle switch, 315
 Logic analyzers, 16
 Long data types, 559
 Looping, 545
 Lowe's, 17
 LucasArts, 281
 Lucent Rigs, 82
 Lukasz Brunn's Mouthshut Web site, 176
 Luminance (luma), 364, 382

M

Macintosh file system, 590
 Macintosh OS X, 166, 474
 Magazines, 537
 MagicGate, 165
 Mainboard, PS2. *see* PlayStation 2 (PS2), installing serial port in
 MAME (Multiple Arcade Machine Emulator), 244
 Mapper 0 (NROM) game cartridges, 324–325, 330
 Mappers, 330, 332–333
 Marble Craze game, 392
 Marking, IEC, 522–523
 Mars Pathfinder, 599
 Masking tape, 7
 Mathematical functions in C, 559–562
 Maxim Web site, 157
 MB (megabytes), 588
 MCM Electronics, 316
 McMaster-Carr, 17
 MediaPlayer, PS2Reality group, 165, 168
 Megabytes (MB), 588
 Memory
 cache, 591–592
 management, 588–589, 599
 physical, 587–589
 unallocated, 571–572, 576
 virtual, 589–590
 Memory architecture of GBA, 228–229
 Memory card adapters, USB, 168
 Memory card in Xbox controller, 107–110

MESS (Multiple Emulator Super System), 506
 Metal oxides, 534
 Microsoft Xbox. *see* Xbox
 Microsoft Xbox Wireless Adapter, 124
 Midrange tools, 8–12
 Mini-ITX form factor, 28
 Minimum specifications
 Windows 2000/XP, 4
 Minix, 598
 Minus sign (–), 561
 Mistakes, common, 536–537
 Modchips
 PlayStation 2 (PS2), 148–150
 Xbox, 131–141
Modern Operating Systems (book), 594, 601
 Modes of GBA, 232
 Momentary switches, 39–40
 Monitor cartridge, 7800/2600, 503
 Monkey Island game, 281
 Moon Patrol game, 419
 Motherboards
 Atari 2600, 70–71
 configuring, 35–37
 VIA Technologies EPIA Nehemiah M10000, 24–25, 28–29
 vias (holes), 160
 see also Mainboard *under*
 PlayStation 2 (PS2), installing serial port in
 Mountain King game, 413
 Mouse receiver, cordless, 26, 46–50, 74, 77
 Mouser Electronics, 539
 Mouthshut Web site, 176
 Movies, streaming, 168
 MP3s, streaming, 168
 Mr. Spiv's Multifirmware (MultiFW2), 276, 282
 Multiboot cable, GBA, 233–234
 Multics, 597
 Multifirmware loader, installing, 275–280
 Multimeters, 13
 Multiple Arcade Machine Emulator (MAME), 244

Multiple Emulator Super System (MESS), 506
 Mylar sheet, traces on, 425

N

Napalm development team, 169
 Naplink USB boot loader, 165
 Needle files, 6–7
 Needle-nose pliers, 10
 Nehemiah M10000 motherboard, 24–25, 28–29
 Neonode N1 “limitless” mobile device, 600
 NES. *see* Nintendo Entertainment System (NES 1 or Toaster); Nintendo Entertainment System (NES 2 or Top Loader); Nintendo Entertainment System (NES) game cartridges
 NES 3.8mm security bit (gamebit), 316, 318
 NES City Web site, 333
 NES control pad. *see* Nintendo Entertainment System (NES) control pad
 NES Mappers Web site, 330
 NES Player Web site, 333
 “NES Seal of Approval,” 311
 NES
 Technical/Emulation/Development FAQ, 331
 NES World Web site, 333
 #nesdev IRC Channel, 332
 #nesdev IRC Channel, 332
 NesDev Web site, 331
 NEShq.com Web site, 333
 NESPC Web site, 83
 NetStumbler Forums, 123
 Network Adapter, PS2, 170
 Network status LEDs, adding to Xbox panel, 120–123
 Networks, ad hoc and infrastructure, 124
 New-line character (\n), 557–558
 New NES Forums Web site, 333
 Newark Electronics, 539
 Nibbles, 510–511
 Nibbling tools, 10

Nintendo. *see* Game Boy Advance (GBA); Game Boy models; Nintendo Entertainment System (NES 1 or Toaster); Nintendo Entertainment System (NES 2 or Top Loader); Nintendo Entertainment System (NES) control pad; Nintendo Entertainment System (NES) game cartridges; Nintendo Famicom Console
 Nintendo corporate history, 186–187
 Nintendo e-Reader, 235–237
 Nintendo Entertainment System (NES 1 or Toaster)
 “Bratwurst Toaster,” 332
 cartridge connector, 300
 EPROM development cartridge, 329–332
 failure, most common, 300
 history, 292
 illustration, 293
 other hacks, 332–333
 power LED, replacing, 302–310
see also Lockout Chip, NES
 Nintendo Entertainment System (NES 2 or Top Loader)
 blue power LED modification, 302–311
 cartridge connector, 300
 cartridge-loading mechanism, 297–298
 composite audio/video output, 332
 EPROM development cartridge, 329–332
 flashing blue screen, 299, 302, 315
 history, 292
 illustration, 294
 Lockout Chip, absence of, 312
 power LED, absence of, 302
 reassembly, 299, 310, 314–315
 Nintendo Entertainment System (NES) control pad
 Atari 2600, 356–363
 Atari 7800, 502
 reassembly, 363
 wire colors, 362

Nintendo Entertainment System (NES) game cartridges
 battery, replacing, 319–323
 case, opening, 316–319
 EPROM cartridge, creating, 324–330, 332
 game development, homebrew, 324–332
 hacks, other, 332–333
 reassembly, 319
 Nintendo Famicom Console
 history, 292
 illustration, 293
 tribute site, 333
 Nintendo Game Boy. *see* Game Boy Advance (GBA); Game Boy models
 Nintendo Land Web site, 333
 Nowak, Christian, 282
 NPN transistors, 527
 nPort utility, 169
Nuts & Volts magazine, 537

O

Object Attribute Memory (OAM) of GBA, 231
 Obsolete components, finding, 539
 Official Xbox Developer Information Web site, 145
 Official Xbox Magazine Web site, 146
 Official Xbox Site, 146
 Ogg Vorbis files, streaming, 168
 Ohm, George Simon, 518
 Ohm’s Law, 311, 486, 518–519
 Ω (resistance symbol, omega), 519
 Online Technology Exchange, 539
 The Open Group, 597
 Open Source Initiative (OSI), 597
 Open-source software, 597
 OpenXDK Web site, 145
 Operands in assembly language, 581
 Operating systems
 block devices, 595–596
 character devices, 594–595
 description, 586
 device drivers, 593–594
 dining philosophers problem, 593
 embedded systems, 599–600

file systems, 590–591
 GUIs, 593
 input/output (I/O), 592
 Linux, 596–597
 memory, cache, 591–592
 memory, physical, 587–589
 memory, virtual, 589–590
 processes, 592
 references, 594, 596, 601
 shells, 593
 system calls, 593
 user interfaces, 593
 VxWorks, 599
 Windows CE, 599–600
 Operations in assembly language, 580
 Operators
 comparison, 564
 increment/decrement, 564
 other, 575
 Oscilloscopes, 15
 OSI (Open Source Initiative), 597
 Output

Atari 2600 (VCS) to TV, 37
 input/output (I/O), 592
 streams, 578
see also Composite audio/video output; Serial input/output (SIO)

Output streams, 578
 Overclocking, 264
 Overhead lighting, 5
 Oxides, metal, 534

P

Painting, 20
 Palette memory of GBA, 230
 Panasonic Fans Web site, 83
 Parallel
 capacitors in, 523
 resistors in, 520
 Parallel architecture, 171
 Part designators, 514
 Partitioning hard drives, 36
 PartMiner Web site, 539
 Parts and materials
 Atari 2600 (VCS), 83
 Passing variables, 573–574

- Pathfinder, Mars, 599
 PCB. *see* Printed-circuit boards (PCBs)
 PDRoms Web site, 283
 Peak voltage (V_{PEAK}), 517
 Personal-injury warnings. *see* Safety warnings
 Philosophers, dining, 592
 Phone jack, 498
 Photodiode designators and symbols, 515
 PimpRig: Smack Ya Rig Up! Web site, 82
 Pinouts and signals
 Atari 2600 solder pad, 372
 GBA Stealth Dimmer Chip pinout, 225–226
 PlayStation 2 (PS2), 149
 RJ-45 plug, 119
 s-video connector, 372–373
 voltage regulators, 494
 Pitfall game, 413
 Pitfall II game, 391
 Pixels Past, 394
 Plastic header, 5-pin, 162
 Plastic posts, 404
 PlayStation 2 (PS2)
 case, opening, 152–156
 Emotion Engine (EE), 156, 164, 171–175
 game development, homebrew, 176
 I/O processor (IOP), PS2, 175–176
 Linux Kit, 170, 173, 177
 mainboard identification, 151–152
 mainboard revisions, 150–151
 modchips, 148–150
 Network Adapter, 170
 pinout, 149
 PS2Lib Open Source library, 150
 reassembly, 162
 source code examples, 150
 streaming movies, MP3s, and Ogg Vorbis files, 168
 technical details, 171
 versions, 150–152
- see also* PlayStation 2 (PS2), booting from memory card; PlayStation 2 (PS2), installing serial port in
 PlayStation 2 (PS2), booting from memory card
 BOOT.ELF file, 168
 description, 164
 explanation of hack, 169–170
 preparation for hack, 165
 TITLE.DB file, preparing, 165–168
 TITLE.DB file, saving to memory card, 168–169
 PlayStation 2 (PS2), installing serial port in
 description, 156
 explanation of hack, 164
 mainboard, identification of, 151–152
 mainboard, modifying, 158–163
 mainboard, revisions of, 150–151
 preparation for hack, 157–158
 SIO cable, 157
 testing, 164
 tools, 157–158
 PlayStation PC Web site, 83
 PLCC, removing, 533
 Plexiglas windows, 20
 Pliers, needle-nose, 10
 Plus sign (+), 561
 PNP transistors, 527
 Pointers, 567–570
 POKEY chip, 463–465, 479
 Portable game systems, resources for, 83
 Posts, plastic, 404
 Potential difference, 516–518
 Potentiometers (variable resistors), 350, 433, 514
 Power, 516–517
 Power connectors
 Atari 2600, 40–42, 64–65
 Atari 7800, 495
 ATX, 56–59
 phone jack, 498
 see also Connectors
 Power Extension Cable, ATX, 56–59, 68–69
 Power Glove, 292
 Power jacks. *see* Power connectors
 Power LED modification, GBA, 234–235
 Power Pad, 292
 Power supplies
 adjustable, 13–14
 Atari 2600, 72–73, 496
 modifying, 20
 Power Supply Module, ATX, 54–55
 Power supply plug retrofit, 495–501
 Power switch, Atari 2600, 76
 Preparations for hacks
 accelerating GP32, 265
 Atari 2600, converting to PC, 23–29
 Atari 2600 audio/video modification, 384
 Atari 2600 joystick hacks, 338, 342
 Atari 2600 paddle maintenance, 350
 Atari 2600 s-video/audio modification, 364–365
 Atari 5200, opening, 401
 Atari 5200 BIOS replacement, 414
 Atari 5200 blue LED modification, 409
 Atari 5200 four-port switchbox, 435
 Atari 5200 paddle controller, 421
 Atari 7800 ProSystem, blue LED modification, 480
 Atari 7800 ProSystem, game compatibility hacks, 487
 Atari 7800 ProSystem, power supply plug retrofit, 496
 Atari 7800 ProSystem, voltage regulator replacement, 490
 DC power adapter, creating, 269–270
 EPROM cartridge, creating, 324–325
 game cartridge battery, replacing, 316, 320
 GBA Afterburner kit, installing, 198–200
 GBA case, opening, 187
 GBA display lens, replacing, 194

- Linux, installing in Xbox, 141
 modchip, installing in Xbox, 135
 Multifirmware loader, installing, 275–280
 NES 1 Lockout Chip, disabling, 312
 NES 2 blue power LED modification, 303
 NES 72-pin cartridge connector, replacing, 300
 NES control pad with Atari 2600, 357
 NES game cartridge, opening, 316
 network status LEDs, adding to Xbox panel, 120
 PlayStation 2 (PS2), booting from memory card, 165
 PlayStation 2 (PS2), installing serial port in, 157–158
 removable antenna, adding to Xbox, 126
 Xbox controller, adding remote reset switch, 104–105
 Xbox controller, illuminating buttons, 99
 Xbox controller, opening, 97
 Xbox controller memory card, adding remote reset switch, 107–108
 Xbox crossover cable, making, 117
 Xbox Live Communicator, adding, 111–112
 Preprocessor commands, 577–578
 PRG-ROM image, 328, 330
 Printed-circuit boards (PCBs)
 etching kits, 15
 soldering to, 530–532
printf command, 557–558, 561, 576–578
 Printing to the screen in C, 556–558
 Processes
 description, 592
 interrupts, 598
 Programming concepts
 arrays, 549–550, 567–570
 assignment, 543–544
 comments, 553–554, 560
 conditional branching, 545–546
 control structures, 544–548,
 562–567
 debugging, 575–578
 formatting code, 555
 function and variable names, 554
 hash tables, 551
 languages, 542
 linked lists, 551–553
 looping, 545
 preprocessor commands, 577–578
 readability, 553–555
 references, 584
 structures, 549, 572–573
 unconditional branching,
 546–548
 variables, 543
Programming from the Ground Up
 (book), 584
 PROM burners, 14–15
 Protective gear, 5
 PS/2 connector, 49
 PS1 games, title IDs for, 166–167
 PS2. *see* PlayStation 2 (PS2)
 PS2 Independence exploit. *see*
 Independence exploit, PS2
 PS2 Scene Forums Web site, 177
 ps2dev Web site, 176
 PS2Emu Web site, 177
 PS2Lib Open Source library, 150
 ps2link Network Adapter loader,
 165
 PS2Reality group MediaPlayer,
 165, 168
 Pukklink Network Adapter loader,
 165
 Punches, center, 10
 PW70 ATX Power Supply Module
 for Atari 2600, 54–55, 72–73
- ## Q
- QFP, removing, 533
 Quattro Adventure game, 312
 Question mark (?), 558
- ## R
- Radio frequency (RF) shielding, 43
 conditional branching, 545–546
 control structures, 544–548,
 562–567
 debugging, 575–578
 formatting code, 555
 function and variable names, 554
 hash tables, 551
 languages, 542
 linked lists, 551–553
 looping, 545
 preprocessor commands, 577–578
 readability, 553–555
 references, 584
 structures, 549, 572–573
 unconditional branching,
 546–548
 variables, 543
Programming from the Ground Up
 (book), 584
 PROM burners, 14–15
 Protective gear, 5
 PS/2 connector, 49
 PS1 games, title IDs for, 166–167
 PS2. *see* PlayStation 2 (PS2)
 PS2 Independence exploit. *see*
 Independence exploit, PS2
 PS2 Scene Forums Web site, 177
 ps2dev Web site, 176
 PS2Emu Web site, 177
 PS2Lib Open Source library, 150
 ps2link Network Adapter loader,
 165
 PS2Reality group MediaPlayer,
 165, 168
 Pukklink Network Adapter loader,
 165
 Punches, center, 10
 PW70 ATX Power Supply Module
 for Atari 2600, 54–55, 72–73
- ## R
- Radio frequency (RF) shielding, 43
 Radio Shack
 books, 537
 source for tools, 17
 Random access memory (RAM),
 587
 Readability, 553–555
 Real-time operating systems
 (RTOS), 599
 Reassembly
 Atari 2600 (VCS), 67–78,
 379–380, 390
 Atari 5200 Paddle Controller, 431
 Atari 5200 SuperSystem, 408
 Atari 7800 ProSystem, 484, 489,
 494, 500
 Atari CX-40 joystick, 341,
 347–348
 Game Boy Advance (GBA), 193
 GamePark 32 (GP32), 257
 NES 2, 299, 310, 314–315
 NES control pad, 363
 NES game cartridge, 319
 PlayStation 2 (PS2), 162
 Xbox controller, 103
 Xbox controller memory card,
 110
 Xbox Live Communicator, 110
 Xbox Wireless Adapter, 129
 Register map, SIO, 173
 Registers of GBA, 227–228
 Registration process for GP32, 245
 Remote reset switch
 adding to Xbox controller,
 104–107
 adding to Xbox controller
 memory card, 107–110
 Removable antenna, adding to
 Xbox, 125–131
 Reset switch
 adding to Xbox controller,
 104–107
 adding to Xbox controller
 memory card, 107–110
 Resistance (Ω)
 description, 518
 symbol, 519
 Resistors
 current-limiting, 310–311,
 485–486, 516, 520, 526

- description, 518–521
 designator and symbol, 514
 divider configuration, 269
 variable (potentiometers), 350, 433, 514
- R**esources
 Atari 2600 (VCS), 82–83, 396–397
 Atari 7800 ProSystem, 506–507
 case modifications, 82
 Game Boy Advance (GBA), 238–239
 GamePark 32 (GP32), 286–287
 portable game systems, 83
 Xbox (XBC), 146
- Respirator, 5
- retrodev.info Web site, 283
- RetroNES Web site, 333
- Reverse recovery time (T_{RR}), 525
- Reverse voltage (V_R), 525
- Rework stations, 532–533
- RF shield, bending, 406
- RF shield, removing
 Atari 2600, 367–370
 Atari 7800, 481–482
- Nintendo Entertainment System (NES 2 or Top Loader), 296–297, 305
- RF standards, 382
- Ripples, 521
- Ritchie, Dennis, 556
- RJ-45 plug, wiring of, 119
- Robot Tank game, 489
- Robotic Operating Buddy (R.O.B.), 292
- Rost, Bob, 331
- Routers and Xbox, 115
- RTOS (real-time operating systems), 599
- S**
S-video
 connector pinout, 372–373
 coupler, 365
 output, 364–380, 501
 standards, 382
- Safety warnings
- personal injury, 4, 38, 44, 59, 89, 92
see also Antistatic precautions; Hardware harm warnings
- Sandpaper, 7
- Saturn PCC Web site, 83
- Saws, 10
- Schell, Chad (Schell's Electronics), 393, 504–506
- Schematics, 514–516
- Screw lengths, 402
- Screwdrivers, 6
- Script Creation Utility for Maniac Mansion (SCUMM), 281
- Seal of Approval, NES, 311
- Security bit, NES 3.8mm (gamebit), 316, 318
- Security driver bits, 9
- Security mechanism, 10NES, 311–312
- Sega Genesis controller modification, 501–502
- Serial input/output (SIO)
 cable, 157
 initialization code, 174–175
 input/output code, 175
 interface board, 163
 port, 156
 register map, 173
- Serial port, installing in PS2. *see* PlayStation 2 (PS2), installing serial port in
- Series
 capacitors in, 523
 resistors in, 520
- Sheff, David, 292
- Shells, 593
- Shield, connector, 62
- Shielding, radio frequency (RF), 43
- Shockley, William, 526
- Short data types, 559
- SIF (sub-CPU Interface), 176
- Signature Key Generator, Atari 7800, 505
- Signed data types, 559
- Silencing, 20
- Single quotation mark ('), 558
- SIO. *see* Serial Input/Output (SIO)
- Skeleton+ game, 391
- Skin, custom, for GBA, 237–238
- Slalom game, 324
- Slash character (/), 561
- Smackdown GT Web site, 333
- SmartMedia external memory card, 243
- SMDs (surface-mount devices), 532–536
- Smocks, 5
- SoC (System-on-a-Chip) processors, 156, 171
- Software, installing in Atari 2600 (VCS), 36–38
- SOIC, removing, 533
- Solder pads
 pinouts and signals, 372
 removing solder from, 327
 warnings, 266, 306, 308, 325, 361, 410, 483
- Solder sucker, 532
- Soldering
 accessories, 11–12
 desoldering, 532–536
 examples, 530–532, 533–536
 faulty connections, 536
 Improper handling, 530
 stations, 11
 techniques, 530–536
see also Desoldering; Solder pads
- Sony PlayStation 2 (PS2). *see* PlayStation 2 (PS2)
- Soper, John, 502
- Sound system of GBA, 231
- Source code examples, 150
- Sources for tools, 16–17
- Space Shuttle game, 489
- Speakers
 Atari 2600, 382
 designators and symbols, 515
- Spiv's Multifirmware (MultiFW2), 276, 282
- Sponge, 532
- Spray Paint Your GP32 Web site, 284
- ST utility, Atari, 503
- Stacks in assembly language, 583–584
- Stallman, Richard, 597–598
- Standard library *stdio.h*, 558

- StarFire game, 394–395
 Starpath, 393
 Start and Select Button LED Mod Web site, 284
 Static discharge protection. *see* Antistatic precautions
 Stations, rework, 532–533
stderr (standard error) and *stdout* (standard output), 578
stdio.h standard library, 558
 Stealth Dimmer Chip
 description, 217
 installing, 218–225
 pinout, 225–226
 preparation for installing, 218–219
 tools for installing, 218–219
 Stella Mailing List, 395
 Stelladaptor 2600 Controller-to-USB Interface, 22–24, 51–53, 74, 78
 Stereo audio output, Atari 2600, 382–391
 Stolberg, Eckhard, 502, 505
 Storage structures
 arrays, 549–550, 567–570
 description, 548–549
 hash tables, 551
 linked lists, 551–553
 pointers, 567–570
 structures, 549, 572–573
 unallocated memory, 571–572, 576
 Streaming movies, MP3s, and Ogg Vorbis files, 168
 Streams, output, 578
 Strings
 in assembly language, 577, 581
 in C, 571–572
 description, 559
 Strippers, wire, 10
 Strongly typed language, 558
Structured Computer Organization (book), 584
 Structures, 549, 572–573
 Structures, control, 544–548, 562–567
 Sub-CPU Interface (SIF), 176
 Sucker, solder, 532
 Super Breakout game, 419
 Super Mario Brothers game cartridge, 292, 324
 Supercharger peripheral, 393, 489
 Surface metal oxides, 534
 Surface-mount devices (SMDs), 532–536
switch statements, 566–567
 Switches
 Atari 2600 (VCS), 22, 30, 76
 designators and symbols, 515
 desoldering, 38–39
 momentary, 39–40
 Symbolic constants, 562
 Symbols, schematic, 515–516
 System calls, 574, 593
 System-on-a-Chip (SoC)
 processors, 156, 171
 SYSTEM.CNF file, 166–167, 169
- ## T
- tamatypes.h header, 150
 Tap Plastics Web site, 83
 Tape
 adhesive, 7
 foam, 71, 79
 τ (time constant, tau), 524
 TeamXbox Web site, 146
 Techniques for soldering, 530–536
 Tecmo Super Bowl game, 319
 Television Interface Adapter (TIA)
 chip, 375–377, 385–386, 389, 391
 Televisions
 jacks, 364
 standards, 381–382
 10 Yard Fight game, 324
 Test Equity, 17
 Test mode, GamePark 32 (GP32), 284–286
 The Dig game, 281
 The Third Creation Web site, 176
 Thompson, Kenneth, 556
 Thrust+ Platinum game, 392
 Thumb drives, USB, 142
 TIA (Television Interface Adapter)
 chip, 375–377, 385–386, 389, 391
 Tile-based modes of GBA, 232
 Tilton, Jay, 501
 Time constant (τ), 524
 Title IDs for PS1 games, 166–167
 TITLE.DB file
 loading, 169–170
 preparing, 165–168
 saving to memory card, 168–169
 titleman utility, 165–168
 tniNES ROM Editing Utility, 331
 Toaster, Nintendo. *see* Nintendo Entertainment System (NES 1 or Toaster)
 Tom's Hardware Forums, 123
 Tools
 advanced, 13–16
 debugging, 575–576
 essential, 5–8
 midrange, 8–12
 overview, 4
 sources for, 16–17
 Tools for specific hacks
 accelerating GP32, 265
 Atari 2600, converting to PC, 26–27
 Atari 2600 audio/video modification, 384
 Atari 2600 joystick hacks, 338, 342
 Atari 2600 paddle maintenance, 350
 Atari 2600 s-video/audio modification, 365
 Atari 5200, opening, 401
 Atari 5200 BIOS replacement, 414
 Atari 5200 blue LED modification, 410
 Atari 5200 four-port switchbox, 436
 Atari 5200 paddle controller, 421
 Atari 5200 video and audio upgrade, 448
 Atari 7800 blue LED modification, 480
 Atari 7800 game compatibility hacks, 487
 Atari 7800 power supply plug retrofit, 497
 Atari 7800 voltage regulator replacement, 491
 DC power adapter, creating, 271

GBA Afterburner kit, installing, 198
 GBA case, opening, 187
 GBA display lens, replacing, 194
 modchip, installing in Xbox, 135
 NES control pad with Atari 2600, 357
 network status LEDs, adding to Xbox panel, 120
 PlayStation 2 (PS2), installing serial port in, 157–158
 PlayStation 2 (PS2), opening, 152
 removable antenna, adding to Xbox, 126
 Xbox controller, adding remote reset switch, 104
 Xbox controller, illuminating buttons, 99
 Xbox controller, opening, 97
 Xbox controller memory card, adding remote reset switch, 107–108
 Top Loader. *see* Nintendo Entertainment System (NES 2 or Top Loader)
 Torvalds, Linus, 598
 Traces, cutting, 406
 Traces on Mylar sheet, 425
 Transistors
 description, 526–528
 designators and symbols, 515
 orientation, 537
 T_{RR} (reverse recovery time), 525
 TSOP, removing, 533
 Tubing, heat-shrink, 9
 TV output, 37
 Two-port BIOS replacement for Atari 5200, 413–419
 TX79 processor, 171, 173

U
 UCLinux (embedded Linux), 598
 Ultima: Exodus game, 319
 Ultimate Computer Case Mod Web site, 82
 Unallocated memory, 571–572, 576
 Unconditional branching, 546–548
 Universal Serial Bus (USB) thumb drives, 142

University of California at Berkeley, 597
 University of Washington EE Circuits Archive Web site, 538
UNIX
 Game Boy Advance (GBA), 235
 history of, 597
 Unsigned data types, 559
USB
 backplane, preparing, 45–46
 components, Atari 2600, 73–75
 ports, 29
 USB memory card adapters, 168
 USB-to-Xbox interface, 144
 USBid, 539
 User interfaces, 593
 Utilities
 nPort, 169
 titleman, 165–168
 UV-erasable EPROM devices, 15, 324

V

V-I curve, diode, 525
 Validation key, 502–503
 Value, passing by, 574
 Van Veen, Nicholas, 165, 168
 Variable and function names, 554
 Variable resistors (potentiometers), 350, 433, 514
 Variable-speed cordless drills, 7
 Variables, 543
 Variables, passing, 573–574
 V_B (base voltage), 526
 Vblanks (vertical blanks), 176
 V_C (collector voltage), 526
 V_{CORE} , 160–161
 VCS, Atari. *see* Atari 2600 (VCS)
 V_E (emitter voltage), 526
 Vendel, Curt, 504
 Versions of PlayStation 2, 150–152
 Vertical blanks (Vblanks), 176
 VIA Arena, 82
 VIA Technologies
 EPIA Nehemiah M10000 motherboard, 24–25, 28–29
 Mini-ITX motherboard, 82
 Web site, 28, 35

Vias (holes), 160
 Video 61 (Harry Dodgson)
 7800/2600 Monitor Cartridge, 503
 Video Computer System, Atari. *see* Atari 2600 (VCS)
 Video RAM (VRAM) of GBA, 230
 Video standards, 381–382
 Virtual reality control, 292
 Virtual Super System Web site, 474
 Voltage, 516–518
 Voltage regulators
 linear, 495
 pinout, 494
 replacement, Atari 7800 ProSystem, 490–495
 V_{PEAK} (peak voltage), 517
 V_R (reverse voltage), 525
 VxWorks, 599

W

Wait states, 229
 Warnings, safety. *see* Hardware harm warnings; Safety warnings
 The Warp Zone Web site, 333
 Washington, University of, EE Circuits Archive Web site, 538
 Watts, 516
 Waveforms, 517
 Wearable Internet Appliance (WIA), Hitachi, 600
 Web sites
 2600 101, 396
 2600 Programming for Newbies, 395–396
 5200BAS Compiler, 474
 accupc.com, 83
 Advanced Atari Gaming Studies, Institute for, 506
 AnimalSNES, 83
 Atari 800 ITX, 83
 Atari 2600, resources for, 82–83
 Atari 2600 FAQ, 502
 Atari 2600 Programming Resources, 396
 Atari 7800 Tech Page, 505
 Atari History Museum, 396–397, 475, 504, 506

- Atari Times, 397
 Atari800, 474
 Atari800MacX, 474
 Atari800Win PLus, 474
 Atari7800.org, 506
 AtariArchives.org, 474
 AtariAge, 394–396, 473–474, 506
 AtariMagazines.com, 474
 AtariProtos.com, 397, 475,
 506–507
 bank switching, 487
 B&C ComputerVisions, 397, 475,
 507
 Best Electronics, 82, 83, 342, 397,
 475, 507
 Biidots!, 28
 Bigass NES Mapper List, 330
 “Bratwurst Toaster,” 332
 Case Mod Supplies, 83
 ChipQuik SMD Removal Kit,
 533
 Cobbleware, 264
 Composite Audio/Video Output
 for NES 2, 332
 Comprehensive NES Mapper
 Document, 330
 connector adapters, 502
 Cxbx, The Xbox Emulator, 145
 CyberTech, 364
 Dan B’s Atari 5200 Tech Page,
 473
 Dan Peori, 176
 Dark Fader’s DumpFW, 276
 Dark Fader’s Gamepark 32, 283
 DASM Assembler, 396
 DevKitAdvance software
 development kit, 281
 DevOS modification and cable
 creation for Atari 7800, 502
 Digi-Key Corporation, 83
 Directron.com, 83
 Discover Circuits, 538
 Dreamcast PC, 83
 eBay, 28
 EE Circuits Archive, 538
 EE Compendium, 538
 electrical engineering resources,
 538
 Electro Tech Online, 538
Embedded Linux Journal, 601
 Embedded.com, 601
 ePanorama, 538
 EPROM cartridges, creating, 332
 European GP32 Site, 287
 Exploit Installer, 165
 FireWire Depot, 83
 fr★shgloop, 145
 game development, homebrew,
 145, 176, 331
 game development, university
 course in, 331
 Gamepark Holdings, 287
 Gamepark32.co.uk, 287
 GameSpy GBA Forums, 238
 GBAX.com, 239, 287
 GDB debugger, 156
 GeePee32 Emulator, 283
 GNU Project, 597
 GP32 add-on package, 282
 GP32 Devr’s, 283
 GP32 FLU Dimmer Chip Hack,
 284
 GP32 GCC Toolchain in Linux,
 283
 #GP32 IRC Channel, 287
 GP32 JTAG Adapter, 284
 GP32 World, 287
 GP32 Xtreme, 287
 #gp32dev IRC Channel, 283
 gp32dev Yahoo! Group, 283
 GP32Emu, 283
 GP32Linux, 284
 GP32Linux gP32oPie pre-alpha,
 284
 GP32Linux HOWTO, 284
 GPAdvance, 283
 gpQuake, 283
 Graybill, Mark, 501
 IC Master, 538
 IGN Game Boy, 239
 Independence, PS2, 165
 Integrated Circuit Identification
 (IC-ID), 539
 Jnes NES Emulator, 331
 JoyGP, 287
 Jum52, 474
 JunKmachine, 333
libHDD, 171
 Lik Sang, 287
 Linux Devices, 601
 Loading Linux onto the GP32,
 284
 Lucent Rigs, 82
 Lukasz Brunn’s Mouthshut, 176
 magazines, 537
 mappers, 333
 Maxim, 157
 MCM Electronics, 316
 Mr. Spiv’s Multifirmware
 (MultiFW2), 276, 282
 Multiple Emulator Super System
 (MESS), 506
 Naplink USB boot loader, 165
 NES City, 333
 NES Mappers, 330
 NES Player, 333
 NES
 Technical/Emulation/Development FAQ, 331
 NES World, 333
 NesDev, 331
 NEShq.com, 333
 NESPC, 83
 NetStumbler Forums, 123
 New NES Forums, 333
 Nintendo Europe, 238
 Nintendo Game Boy Advance,
 238
 Nintendo Land, 333
 nPort utility, 169
 Official Xbox Developer
 Information, 145
 Official Xbox Magazine, 146
 Official Xbox Site, 146
 OpenXDK, 145
 Panasonic Fans, 83
 PartMiner, 539
 PDRoms, 283
 PimpRig: Smack Ya Rig Up!, 82
 PlayStation 2 (PS2) Linux Kit,
 177
 PS2 Scene Forums, 177
 ps2dev, 176
 PS2Emu, 177
 ps2link Network Adapter loader,
 165
 PS2Reality group MediaPlayer,
 165, 168
 Pukklink Network Adapter
 loader, 165

removable antenna, adding to Xbox, 125
 retrodev.info, 283
 RetroNES, 333
 routers and Xbox, 115
 Saturn PC, 83
 security driver bits, 9
 smackdown GT, 333
 Spray Paint Your GP32, 284
 Start and Select Button LED Mod, 284
 Stella Mailing List, 395
 Stereo Audio Output Modification, 332
 Supercharger modification, 393
 Tap Plastics, 83
 TeamXbox, 146
 The Third Creation, 176
 Tilton, Jay, 501
 tniNES ROM Editing Utility, 331
 Tom's Hardware Forums, 123
 Ultimate Computer Case Mod, 82
 VCS Workshop, 502, 505
 VIA Arena, 82
 VIA Mini-ITX motherboard, 82
 VIA Technologies, 28, 35
 video hacks for Atari 7800, 501
 Virtual Super System, 474
 The Warp Zone, 333
 Washington, University of, EE Circuits Archive, 538
 WebEE, 538
 Wind River Systems, 599
 Windows Devices, 601
 wiRe's frame, 176
 X-Factor Development, 145
 Xbox-Hacker, 146
 Xbox-HQ, 146
 Xbox-Saves Manager, 146
 Xbox-Scene, 146
 Xbox versions, 91
 XBOX365, 146
 XboxAddict, 146
 #xboxhacker IRC Channel, 146
 z26 Atari 2600 Emulator, 396
 WebEE Web site, 538
 Weighted dial, 432–433
while loops, 564–565
 White space, 555

WIA (Hitachi Wearable Internet Appliance), 600
 WiFi adapter, adding to Xbox, 123–125
 Wind River Systems, 599
 Windows, Plexiglas, 20
 Windows 2000/XP
 display setting, 37
 minimum specifications, 4
 Windows CE, 599–600
 Windows Devices Web site, 601
 Windows Embedded Device Driver site, 599–600
 Windows file systems, 590
 Windows NT/2000/XP, choice of, 22
 Wire clippers and strippers, 10
 Wire colors
 Atari CX-40 joysticks, 340, 347
 NES control pad, 362
 Wireless bridge, Linksys WET11, 124
 Wireless network adapter, adding to Xbox, 123–125
 wiRe's frame Web site, 176

X

X -ACTO hobby knives, 6
 X-Factor Development Web site, 145
 Xbox controllers
 components, 95
 illuminating buttons, 99–104
 illuminating the logo, 104
 memory card reassembly, 110
 opening, 97–98
 reassembly, 103
 remote reset switch, adding, 104–107
 remote reset switch, adding to memory card, 107–110
 third-party, 97
 versions, 96
 see also Live Communicator, Xbox

Xbox-Hacker Web site, 146
 Xbox-HQ Web site, 146
 Xbox Live BIOS patch, 141
 Xbox-Saves Manager Web site, 146
 Xbox-Scene Web site, 146

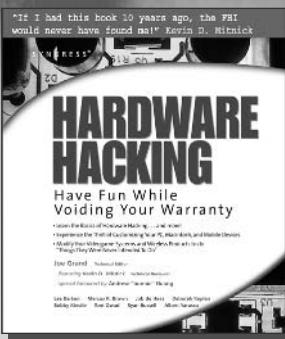
Xbox Wireless Adapter
 adding removable antenna to, 125–131
 adding to Xbox, 123–125
 description, 124–125
 reassembly, 129
 Xbox (XBC)
 case, opening, 92–95
 crossover cable, making, 116–120
 Dashboard, 140
 game development, homebrew, 144–145
 hardware, 89–90
 history, 88
 Linux, installing, 141–144
 manufacturing date, format of, 91
 modchip, installing, 131–141
 network status LEDs, adding to front panel, 120–123
 networking kits, 113
 networking link, establishing, 113–115
 other hacks, 144
 routers, 115
 second hard drive, adding, 144
 serial number, format of, 92
 Universal Serial Bus (USB)
 interface, 144
 versions, 90–91
 Web resources, 146
 Wireless Adapter, adding
 removable antenna to, 125–131
 wireless network adapter, adding, 123–125

XBOX365 Web site, 146
 XboxAddict Web site, 146
 #xboxhacker IRC Channel Web site, 146
 #xboxhacker IRC Channel Web site, 146
 Xecuter Lite+ modchip, 132–133, 135–139

Z

z26 Atari 2600 Emulator Web site, 396
 Zapper, 292
 Zelda II: Link's Adventure game, 319

Syngress: *The Definition of a Serious Security Library*



AVAILABLE NOW!
ORDER at
www.syngress.com

Hardware Hacking

Joe Grand, Ryan Russell, Kevin D. Mitnick (Editor)

"If I had this book 10 years ago, the FBI would never have found me!" — Kevin Mitnick

This book has something for everyone--from the beginner hobbyist with no electronics or coding experience to the self-proclaimed "gadget geek." Take an ordinary piece of equipment and turn it into a personal work of art. Build upon an existing idea to create something better. Have fun while voiding your warranty!

ISBN: 1-932266-83-6

Price: \$39.95 US \$59.95 CA

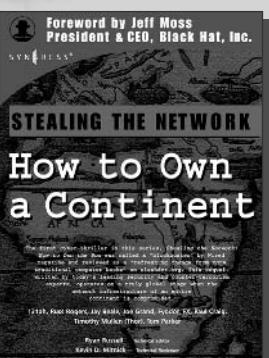
The Mezonic Agenda: Hacking the Presidency

Dr. Herbert H. Thompson and Spyros Nomikos

The Mezonic Agenda: Hacking the Presidency is the first Cyber-Thriller that allows the reader to "hack along" with both the heroes and villains of this fictional narrative using the accompanying CD containing real, working versions of all the applications described and exploited in the fictional narrative of the book. The Mezonic Agenda deals with some of the most pressing topics in technology and computer security today including: reverse engineering, cryptography, buffer overflows, and steganography. The book tells the tale of criminal hackers attempting to compromise the results of a presidential election for their own gain.

ISBN: 1-931836-83-3

Price: \$34.95 U.S. \$50.95 CAN



AVAILABLE NOW!
ORDER at
www.syngress.com

Stealing the Network: How to Own a Continent

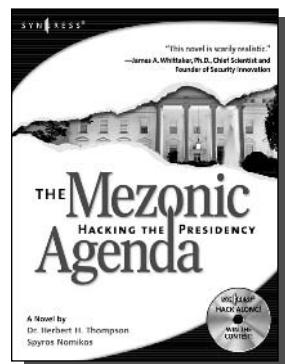
Ryan Russell, FX, Joe Grand, Tim Mullen, Jay Beale, Russ Rogers, Ken Pfeil, Paul Craig, Tom Parker, Fyodor

The first book in the "Stealing the Network" series was called a "blockbuster" by Wired magazine, a "refreshing change from more traditional computer books" by Slashdot.org, and "an entertaining and informative look at the weapons and tactics employed by those who attack and defend digital systems" by Amazon.com. This follow-on book once again combines a set of fictional stories with real technology to show readers the danger that lurks in the shadows of the information security industry... Could hackers take over a continent?

ISBN: 1-931836-05-1

Price: \$49.95 US \$69.95 CAN

Syn•gress (sin-gres): noun, sing. Freedom from risk or danger; safety. See *security*.



SYNGRESS®

TLFeBOOK