
By JOE GRAND

RESEARCH LESSONS FROM HARDWARE HACKING

Want to know how something works? Tear it apart. Along the way, you might learn to improve it or make it do something it was never intended to do.



ardware hacking—modifying a product to do something it was never intended to do by its original designers—and reverse engineering—the art of learning from practical examples—are both important facets in the training of any technology professional. Hardware hackers have a natural curiosity about opening things up and seeing how they work. They then go beyond their glossy, finely manicured shells

to figure out what else can be done with or to improve them.

Although unconventional in the typical educational environment, such activities provide hands-on experience and a look into product design that cannot be learned from a textbook. The hardware hacking community represents an example of nontraditional learning and how the world can be changed for the better by unorthodox thinkers and their experiments (see the sidebar “A Brief History of Hardware Hacking”).



Figure 1. Grand Idea Studio's hardware hacking training course.

The thought processes of hackers and academic researchers are almost mirror images of each other. Hackers tend to learn by looking at a completed technology, taking it apart, working backward by breaking down the system into subsystems, and picking up the theory as needed. This style of learning allows them to discover previous methods of solving particular problems, seeing the solution used in a practical, real-world environment. Along the way, they frequently come up with unexpected uses for or improvements in the technology, depending on the goal of the exercise.

Academic researchers tend to work in the other

direction. In a typical engineering program, theory, including math, physics, and basic electronics, comes first in a classroom setting. Then, only after they have proved themselves proficient in theory, are the students required to use their skills to design and construct some object or project in their final year.

An entertaining hardware hack-turned-academic study is available at the Hacking Microsoft Barney Web site [1]. This group effort by researchers at the Xerox Palo Alto Research Center and Massachusetts Institute of Technology aimed to reverse-engineer the control mechanism of the Microsoft ActiMates Barney plush toy and create a set of software com-

DESIGNERS SHOULD TRY TO BREAK THE SECURITY MECHANISMS OF THEIR OWN PRODUCTS, THEN FIX THEM AND TRY TO BREAK THEM AGAIN.

ponents to allow a user to programmatically control the toy. The resultant “Barney Protocol Stack” allows a user to drive Barney directly via an on-screen control panel to move it around, play sound samples, and read its sensors. Alternatively, the Stack can listen on a network socket for remote-control connections. The remote interface makes it possible for anyone to write applications that talk to a remote Barney server [7].

It appears the desire and opportunity to reverse-engineer some interesting technology—ActiMates Barney—and hack it to do something it was not intended to do by its original developers occurred first; the interesting outcome related to human factors interfaces were a result. This work exemplifies the methodology of combining nontraditional approaches to hardware hacking and a more structured academic environment.

People learn in many different ways, but a hands-on approach helps apply learned academic knowledge to practical problems. Since 2004, I have been leading a two-day training course called “Hands-On Hardware Hacking and Reverse Engineering Techniques” (www.grandideastudio.com/portfolio/index.php?id=1&prod=38) to teach basic electronics, hardware hacking, and reverse engineering skills. I conclude with a full-scale, hands-on hardware hacking challenge in which the students, individually or in small groups, try to defeat the security mechanisms of a custom-designed circuit board and unlock a hidden version of the popular memory game Simon (see Figure 1). Participants report they appreciate being able to experiment with hardware using their own methods, instead of being forced to learn through the more common, theory-based style of academic education.

Hardware hacking and do-it-yourself projects by hobbyists and tinkerers have been popular in recent years primarily due to their free-spirit, unconventional nature, even as the number of graduating engineering students in the U.S. has dropped steadily. The best learning methods depend on individuals’ ways of thinking.

MICROSOFT XBOX

When introduced in 2001, the Xbox was Microsoft’s first video game console, designed to compete directly with the established Sony PlayStation 2 and Nintendo GameCube. The original Xbox employed a PC architecture, making it an attractive target for hardware hackers. The follow-up Xbox 360 was released in the U.S. last November. Although serious hardware hackers began to disassemble the Xbox 360 the day it was released, the copyright protection and authentication schemes have yet to be broken. However, getting past them is likely only a matter of time, as no system is 100% secure, and any system can be broken by sufficiently competent, determined attackers.

The original Xbox platform consisted of several core hardware components: Intel Celeron-class 733MHz processor, nVidia GeForce graphics processing unit (also known as the Northbridge), nVidia media communications processor (also known as the Southbridge), 64MB RAM, 10GB hard drive, DVD drive, and 10/100Mbps RJ45 Ethernet network port. The machine was not dramatically different from a PC circa late 2001 or early 2002, though some components were custom-made for the Xbox (and are comparable to those in PCs).

Andrew “bunnie” Huang’s research with the Xbox began in late 2001 as a simple examination of the console while working on his Ph.D. thesis on super-computer architecture at the Massachusetts Institute of Technology. One of his early interests was learning how Microsoft was authenticating the software running on the machine to prevent execution of unauthorized software, including pirated and homebrew games created by hobbyists and enthusiasts.

Through an extremely detailed and methodological approach, not unlike what you’d see in an academic laboratory, Huang managed to retrieve the sole symmetric 128b encryption key used for RC-4-based protection of a secret boot loader, ultimately allowing him to execute untrusted code on the Xbox. He created a custom tap circuit to intercept data transfer

over the Xbox's HyperTransport bus (an industry-standard, high-speed signal bus between the Northbridge and Southbridge chips). The tap board (see Figure 2) consisted of a single low-voltage differential signaling-to-CMOS logic converter interfaced to a Xilinx Virtex-E field-programmable gate array development board. Xbox security was thus defeated, as it relied on the secrecy of a key Huang figured out how to extract from the hardware.

He then wrote a technical paper at MIT's Artificial Intelligence laboratory [5], made a presentation at the 2002 Cryptographic Hardware and Embedded Systems Conference, created a Web page/blog documenting his technical triumphs and frustrations (Adventures Hacking the Xbox, www.bunniestudios.com/?page_id=5), and ultimately created a seminal book on hardware hacking [6].

Although Microsoft and nVidia (manufacturer of the Northbridge and Southbridge chips whose communications Huang targeted on the HyperTransport bus) were initially displeased with his published work on defeating Xbox security, his results helped both companies understand the motives and threat vectors of potential attackers, leading to a more secure hardware design and authentication scheme on the later Xbox 360 console. The lesson is valuable; a hacker identified a major problem in a mass-market product and, by informing Microsoft and nVidia, likely saved both companies from repeating their mistakes in future products.

Perhaps the only way to stop an attacker is to think like one (see the article "Security Through Legality" by Stephen Bono et al. in this section). Being aware of the latest attack methodologies and trends enables a system's designers to choose the proper means of protection and helps them understand the potential threats and risks against their products. The designers should try

to break the security mechanisms of those products, then fix them and try to break them again. Time should be scheduled for this iterative process during the design cycle. Even more appropriate would be to add an objective third-party viewpoint by hiring a hardware hacker to attempt to defeat the product's security or, as in open source software, release the product into the community before releasing it to the public, letting it be analyzed, taken apart, and modified, aiming to squash bugs or find previously unforeseen faults or limitations.



Figure 2. Andrew "bunnie" Huang's Xbox HyperTransport Bus tap circuit connecting an Xbox and FPGA development board (photo courtesy A. Huang).

LEGAL CONCERNS

Recently enacted laws in the U.S. (such as the Digital Millennium Copyright Act of 1998) have sought to prevent reverse engineering by enabling large corporations to flex their legal and financial muscle against potential competitive threats, as well as against the merely curious. The DMCA was originally intended to stop pirates (selling bootlegged copyright material) from violating copyright laws and defeating anti-piracy protections on copyrighted material. Although used for this purpose, the DMCA's anti-circumvention provisions have also been used to stifle an array of legitimate activities (such as reverse engineering, hardware modification, and public disclosure of potentially dangerous security vulnerabilities in products). Shrink-wrap and other explicit agreements are also used to get users to agree to waive their rights to reverse-engineer, analyze,

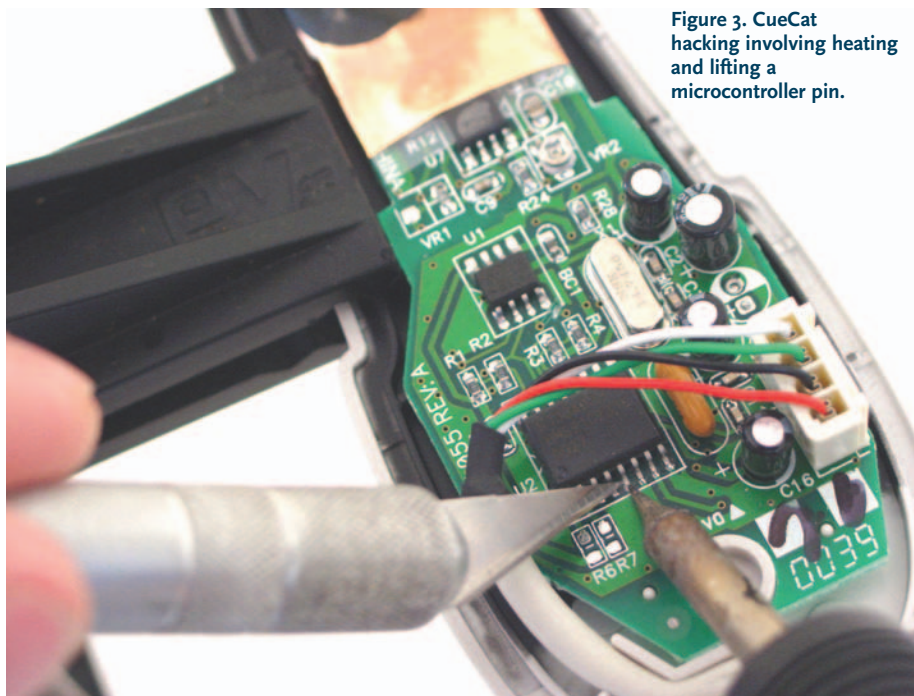


Figure 3. CueCat hacking involving heating and lifting a microcontroller pin.

were violating the company's intellectual property rights. It also claimed that the End User License Agreement included with the CueCat hardware and software stated that CueCat hardware was only on loan to end users who did not own the hardware. DigitalConvergence's claim was challenged by the fact that thousands of end users were sent the CueCat in the form of unsolicited mail, classifying the product as a gift the recipient now legally owned.

Word spread quickly via mainstream media of DigitalConvergence's methods of market research—collecting

or modify products. More commonly used on computer software, these tactics have also been used on hardware. Suddenly, you don't actually own what you've paid for and are reverse engineering.

A classic, fairly mainstream example of when hardware hacking locked horns with unscrupulous legal tactics can be found in the CueCat (see Figure 3) from DigitalConvergence (now defunct), which aimed to integrate classical push media (such as newspapers, magazines, and television) and the Internet. Beginning in 1999, venture capitalists invested more than \$100 million into CueCat technology. The company gave away more than three million CueCat devices free at RadioShack stores throughout the U.S., along with 400,000 to subscribers of *Wired Magazine* and 850,000 to subscribers of *Forbes Magazine* in the mail worldwide. Digital Convergence planned to eventually put 10 million scanners into the public's hands. Using proprietary "cues"—in this case special barcodes—users could swipe cues listed next to products in catalogs they were interested in. When a cue was received, CueCat software on the PC would bring the user to relevant Web pages with more information, usually the Web site of the product or technology.

In August and September 2000, lawyers representing DigitalConvergence sought to use the courts to begin a crackdown on all Web sites that described modifications to CueCat hardware or software, most involving a simple hack to convert the CueCat to read industry-standard barcodes instead of the special CueCat cues. DigitalConvergence claimed that by opening up and modifying CueCat hardware, hackers

and analyzing the CueCat activities of individual end users (unbeknownst to them), including their personal demographics and buying habits, and attempts to squelch the hardware hacking community. Not only was the company's reputation tarnished by the bad press, the technology itself quickly became outdated due to the quick evolution of the Internet. By 2001, the CueCat had been relegated to a historical footnote, and DigitalConvergence was out of business.

Despite these and other legal precedents explicitly protecting and allowing certain types of reverse engineering for personal use or educational/academic purposes, please consult an attorney if you have questions or concerns.

CONCLUSION

I look forward to a future of hardware hacking that stretches the bounds of technology. I also look forward to more academic engineering programs incorporating aspects of hardware hacking, possibly in the form of hands-on, loosely structured exercises. Hardware hacking is an interesting direction professors and developers can incorporate into their daily routines. The do-it-yourself ethos of the hardware hacking community, coupled with the more structured approach of standard academic thinking will continue to lead to new and novel technology. ■

REFERENCES

1. Dourish, P. Exploring software tools for programmable embodied agents, or hacking Microsoft Barney; www.geekchic.com/~jpg/barney.
2. Fullam, S. *Hardware Hacking Projects for Geeks*. O'Reilly Media, Sebastopol, CA, 2003.
3. Grand, J., Yarusso, A., Russell, R., de Haas, J., Brown, M., Barken, L., Owad, T., Kaplan, D., and Kinstle, B. *Hardware Hacking: Have Fun*

While Voiding Your Warranty. Syngress Publishing, Rockland, MA, 2004.

4. Grand, J., Yarusso, A., and Thornton, F. *Game Console Hacking*. Syngress Publishing, Rockland, MA, 2004.
5. Huang, A. *Keeping Secrets in Hardware: The Microsoft Xbox Case Study*. Artificial Intelligence Laboratory Memo 2002-008, Massachusetts Institute of Technology, May 26, 2002; [ftp://publications.ai.mit.edu/ai-publications/2002/AIM-2002-008.pdf](http://publications.ai.mit.edu/ai-publications/2002/AIM-2002-008.pdf).
6. Huang, A. *Hacking the Xbox: An Introduction to Reverse Engineering*. No Starch Press, San Francisco, 2003.
7. Kaminsky, M., Dourish, P., Edwards, W. Keith, LaMarca, A., Salisbury, M., and Smith, I. SWEETPEA: Software tools for programmable embodied agents. In *Proceedings of the Conference on Human Factors in Computing Systems* (Pittsburgh, PA, May 15–20, 1999); www.acm.org/sigchi/chi99/ and sandbox.xerox.com/dourish/sweetpea. pdf.

Several magazines and Web sites also provide useful information on hardware hacking and its colorful community:

MAKE Magazine, www.makezine.com
hack a day, www.hackaday.com
Nuts & Volts Magazine, www.nutsvolts.com
Circuit Cellar Magazine, www.circuitcellar.com

JOE GRAND (joe@grandideastudio.com) is president of Grand Idea Studio, Inc., San Diego, CA.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

© 2006 ACM 0001-0782/06/0600 \$5.00

A BRIEF HISTORY OF HARDWARE HACKING

Hardware hacking dates back almost 200 years. Charles Babbage's Difference Engine in the early 1800s was a mechanical form of hardware hacking. The method William Crookes used to discover the electron in the mid-1800s might have been the first form of electronics-related hardware hacking. Hardware hackers have since been involved in the development of wireless telegraphy, vacuum tubes, radio, television, and transistors. Benjamin Franklin, Thomas Edison, Nikola Tesla, and Alexander Graham Bell were all hardware hackers. So were William Hewlett and David Packard and Steve Jobs and Steve Wozniak.

Contrary to how the word hacker is sometimes used to describe criminals breaking into computer systems, a hacker can be defined more simply as someone involved in the exploration of technology. A "hack" in the technology world usually defines a new and novel creation or method of solving a problem, typically in an unorthodox fashion [2–4]. Here, I focus on an idealistic vision of hardware hackers—the good guys—even though some people, corporations, and agencies may use the same techniques for illegal, illegitimate, or unethical purposes, seeking some sort of financial gain or market advantage.

Hardware hacking means different things to different people, coming down to personal preferences, as in art or music. Someone can be taught, to a point, to have the hacker mindset and break the mold of conventional thinking, but hacking goes much further. It is a passion, a drive, something that stems from some amount of anti-establishment and anti-authority sentiment coupled with the desire to do things on one's own agenda and with one's own hands. Hardware hacking is the technologists' version of the classic phrase "Don't judge a book by its cover." Hackers are driven by a variety of motivations:

Curiosity and fun. See how things work, scratch the curiosity itch, and have fun experimenting with and modifying products;

Education. Learn by doing;

Improvement and innovation. Build a better mousetrap;

Consumer protection. Ensure a product does what its marketing pitch claims it to do. Often distrustful of marketing or sales literature, hackers want to find out for themselves whether certain claims are true and how they can make a particular product do more; and

Security. Test whether hardware devices are secure, identifying failures or weaknesses. Beyond strengthening the perceived value of a product, it allows users to mitigate the risk of an attack by updating, fixing, or discarding the product.

Most hardware hacks fall into four categories:

Personalizing and customizing. Often called "hotrodding for geeks" it includes modifications, custom skins, and even art projects (such as creating an aquarium out of a vintage computer);

Adding functionality. Making the system or product do something it wasn't intended to do (such as converting an iPod to run Linux or modifying a classic Atari 2600 video game console to support stereo sound and composite video output);

Improving capacity or performance. Enhancing or otherwise upgrading a product (such as expanding the recording capacity of a TiVo box by adding a larger hard drive, modifying a wireless network card to support an external antenna, or overclocking a PC's motherboard); and

Defeating protection and security mechanisms. Included are finding "Easter eggs," hidden menus, and backdoors in DVD players or video game consoles or creating a custom cable to unlock the secrets of a cell phone.

Reverse engineering, generally viewed as a subset of hardware hacking, is essentially the art of learning from practical examples and experience. Examining technologies or any kind of product to see how they work is an integral part of the hardware hacking process and is a great way to learn the state of the art. I use reverse engineering to add to my mental toolbox of circuit designs, manufacturing techniques, and printed circuit board layout tricks, all of which improve my knowledge of the product development process. Reverse engineering and hardware hacking represent continuing education, interconnected with developing new products and technologies. **C**