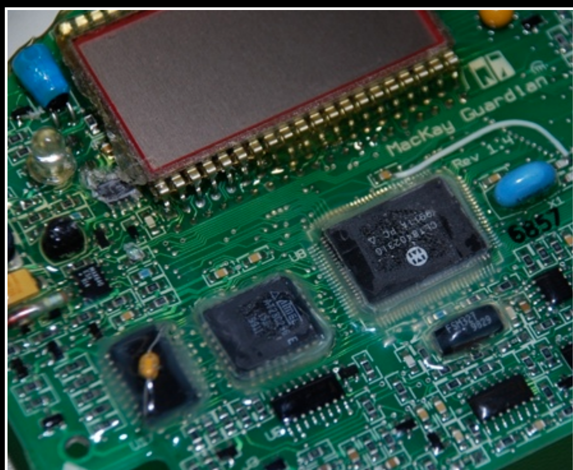
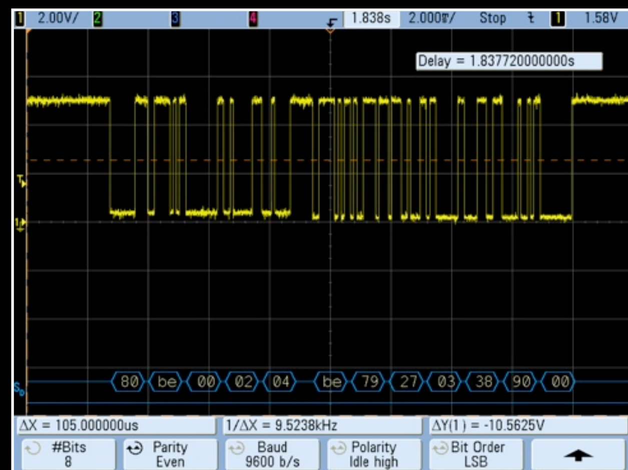
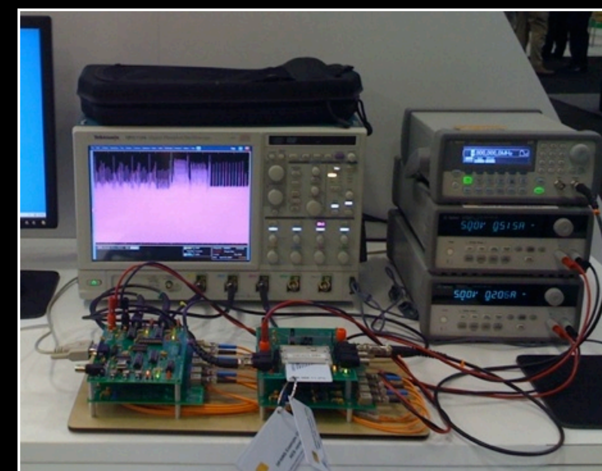
Hardware Reverse Engineering: Access, Analyze, & Defeat







Joe Grand, Grand Idea Studio, Inc.

joe@grandideastudio.com

Black Hat DC 2011 Workshop

Course Outline

(A small taste of my full-fledged HH training)

- Process Overview
- Opening Housings
- Hardware Reverse Engineering
- Memory and Programmable Logic
- External Interfaces
- Advanced Techniques
- Open Lab

We Are Controlled By Technology

- Electronics are embedded into nearly everything we use on a daily basis
- Often taken for granted and inherently trusted
  - H/W is not voodoo, but people treat it that way
- Hardware has largely been ignored in the security field
  - Many products susceptible to compromise via simple, practical classes of attack
  - Vendors mostly respond to security problems by blowing them off (like S/W in the 90s!)
    - ...or it is blown completely out of proportion

Why Hardware Hacking?

- Security competency
  - Test hardware security schemes for failures/weaknesses
- Consumer protection
  - I don't trust glossy marketing materials...do you?
- Military intelligence
  - What is that hardware? How was it designed? By whom?
- Education and curiosity
  - To simply see how things work
  - Do something new, novel, and/or unique

Goals of an Attack

- Theft of service
  - Obtaining a service for free that normally costs $$$
- Competition/cloning
  - Specific theft of information/data/IP to gain a marketplace advantage
- User authentication/spoofing
  - Forging a user's identity to gain access to a system
- Bypass security features/privilege escalation
  - Defeating protection measures or gaining increased control of a system

Common Themes

- Most product design engineers not familiar with security
- Many products based on publicly available reference designs provided by chip vendors
- Components easy to access, identify, and probe
- Engineers and manufacturers want easy access to product for testing and debugging
- Even the simplest attacks can have huge repercussions

Hardware Hacking Methodology

- Major subsystems:
  - Information gathering
  - Hardware teardown
  - Firmware reverse engineering
  - External interface analysis
  - Silicon die analysis

Hardware Hacking Methodology 2

- It's all about gathering clues
- Determination and persistence is the key
  - Keep trying alternative solutions
  - Failure is the most frustrating part of hardware hacking, but also the most educational
  - Don't give up!

Information Gathering

- Crawling the Internet for specific information
  - Product specifications, design documents, marketing materials
  - Check forums, blogs, Twitter, Facebook, etc.
- Acquire target hardware
  - Purchase, borrow, rent, steal, or ask the vendor
  - Ex.: eBay, surplus
- Dumpster diving
- Social engineering

Hardware Teardown

- Hardware and electronics disassembly and reverse engineering
- Get access to the circuitry
- Component and subsystem identification
- Gives clues about design techniques, potential attacks, and system functionality
- Typically there are similarities between older and newer designs
  - Even between competing products

Firmware Reversing

- Extract program code/data from on-board memory devices
  - Using off-the-shelf device programmer or product-specific tool
  - You'll end up with a binary or hex dump
  - Ex.: Flash, ROM, RAM, EEPROM, FPGA
- Now pure software hackers can get into the game
  - Using tools and techniques they are already familiar with
  - Electronic/embedded systems are typically nothing more than a general purpose computer programmed to perform a specific task

Firmware Reversing 2

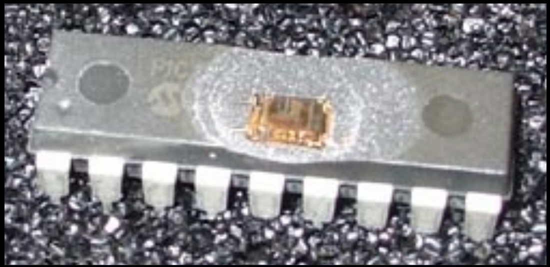
- Quick run through w/ *strings* and hex editor to pick most interesting area to begin with
  - Find clues to possible entry/access points to administrative menus or ideas of further attacks
- Disassemble, modify, recompile, and reprogram device, if desired

Interface Analysis

- Communications monitoring
- Protocol decoding and/or emulation
- Ex.: Smartcard, Serial, USB, JTAG, I2C, SPI, Ethernet, CAN
- Any interface accessible to the outside world may be an avenue for attack
  - Especially program/debug connections: If a legitimate designer has access to the interface, so do we
- Using oscilloscope, logic analyzer, dedicated sniffers, software tools, etc.

Silicon Die Analysis

- Extremely useful depending on attack goals
  - Simple imaging to gather clues
  - Key/algorithm extraction from ICs
  - Retrieve contents of Flash, ROM, FPGAs, other non-volatile devices
  - Cutting or repairing silicon structures (security fuses, traces, etc.)
- Like reversing circuitry, but at a microscopic level



Cracking the Case: Opening Product Housings

Opening Housings: The Basics

- Goal is to get access to internal circuitry
- Have "sacrifical lambs" for initial tests/attempts
- Common case fasteners
  - Screws
  - Plastic snaps molded into case
  - Glue (soften w/ heat gun)
  - Double-sided tape
- Screws are sometimes hidden from the end user
  - On the bottom of the product
  - Under labels
  - Under rubber "feet"

Opening Housings: Step-by-Step

- Prepare a well-lit, clean workbench or area
- Remove power from the device
  - Unplug it, remove batteries, etc.
- Remove any screws (if applicable)
  - Keep track of screw locations if screws are different sizes
  - Store screws in a magnetic bowl or safe place
- Look for seams and gently pull at them
  - Don't force it - the case may be held together by plastic clips
  - Use a small flathead screwdriver or various thickness guitar picks to pry along the seam (if applicable)

Opening Housings: Anti-Tamper Mechanisms

- Physical security for embedded systems
- Attempts to prevent unauthorized physical or electronic tampering against the product
- Most effectively used in layers
- Can almost always be bypassed with knowledge of method
  - Attackers may intentionally destroy a device to determine its security mechanisms

Opening Housings: Anti-Tamper Mechanisms 2

- Resistance
  - Specialized materials used to resist tampering
- Evidence
  - Ensure that there is visible evidence left behind by tampering
  - Only successful if process in place to check for deformity
- Detection
  - Enable the hardware device to be aware of tampering
- Response
  - Countermeasures taken upon the detection of tampering

Anti-Tamper Mechanisms: Do They Work?

- Not really.
- Most seals can be bypassed with ordinary tools
  - Schwettmann & Michaud's "How to Steal Nuclear Warheads Without Voiding Your Xbox Warranty" @ BH DC 2011
  - The Dark Tangent's DEFCON Tamper Evident Contest, `https://forum.defcon.org/forumdisplay.php?f=518`
  - "Vulnerability Assessment of Security Seals," `www.securitymanagement.com/library/lanl_00418796.pdf`

Anti-Tamper Mechanisms: Do They Work? 2

- Argonne National Laboratory, Vulnerability Assessments Team, `www.ne.anl.gov/capabilities/vat/seals/index.html`
- "Potential Chemical Attacks on Coatings and Tamper Evident Seal Adhesives," `http://csrc.nist.gov/groups/STM/cmvp/documents/fips140-3/physec/papers/physecpaper06.pdf`

Results for 244 Seals

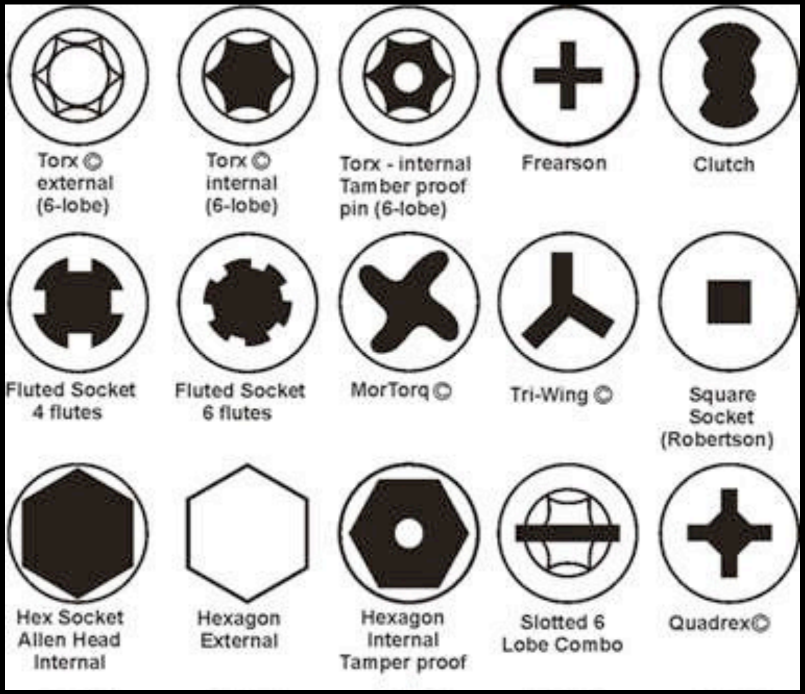
Parameter	Mean	Median
defeat time for 1 person	1.4 mins	43 secs
cost of tools and supplies	$78	$5
marginal cost of attack	62¢	9¢
time to devise successful attack	2.3 hrs	12 mins

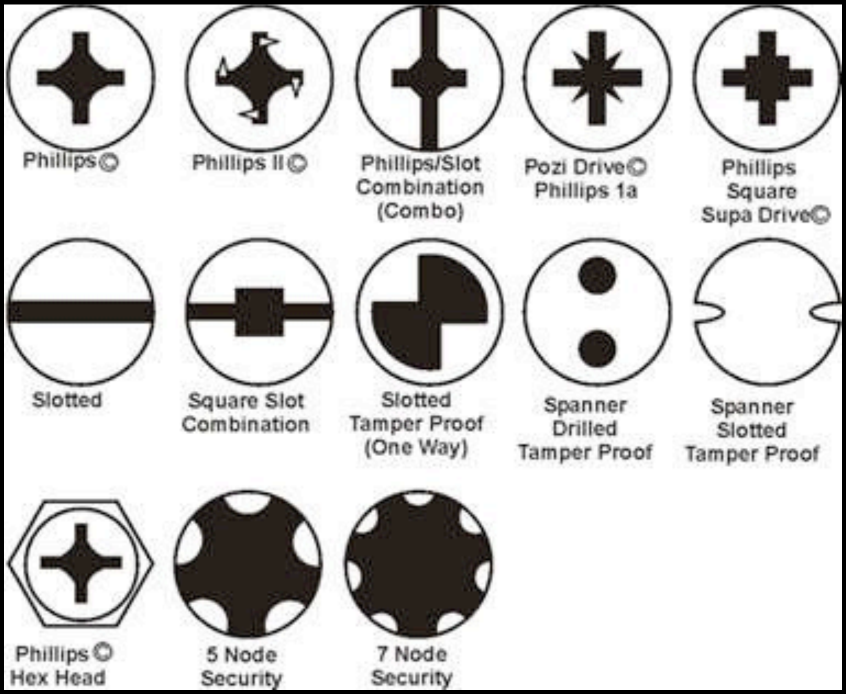
- Half of these seals are in use for "critical" opportunities.
- At least 19% are in use and under consideration for nuclear safeguards.

Opening Housings: Security Bits and One-Way Screws

- Used to prevent housings from being easily opened
- Could be considered an anti-tamper mechanism
- Why are they called "security bits" when you can buy them almost anywhere (or make them)?
  - Ex.: Electronics stores, flea markets, online
  - Only prevents the most simple-minded attackers
- To identify a particular bit type:
  - `www.instructables.com/id/When_a_Phillips_is_not_a_Phillips/`
  - `http://web.archive.org/web/20070806093401/http://www.lara.com/reviews/screwtypes.htm`

Opening Housings: Security Bits and One-Way Screws

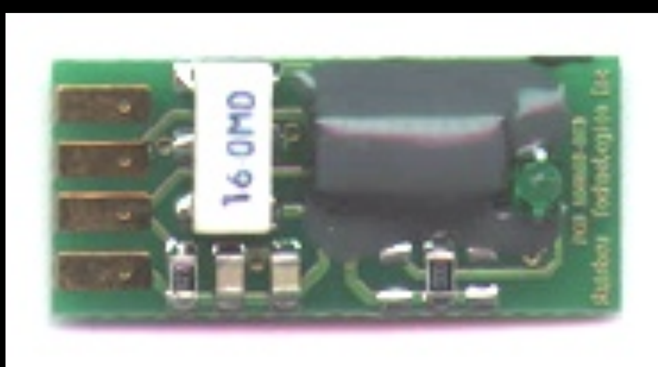


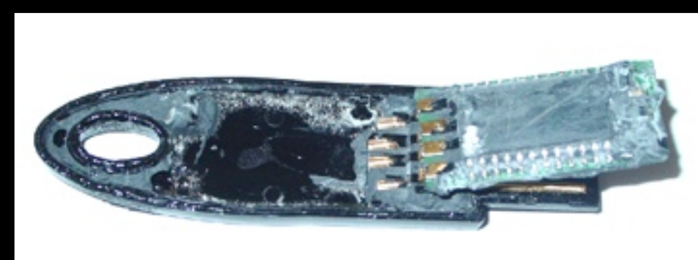


Picture: **www.instructables.com/id/When_a_Phillips_is_not_a_Phillips/**

Opening Housings: Epoxy Encapsulation Removal

- Encapsulation typically used to protect circuitry from moisture, dust, mold, corrosion, or arcing
- Epoxy or urethane coatings leave a hard, difficult to remove film





Opening Housings: Epoxy Encapsulation Removal 2

- The good news: Most coatings are not specifically designed for security...
  - ...though sometimes they're used that way!
- Hot air gun to soften epoxy
- Chemicals
  - MG Chemicals' 8310 Conformal Coating Stripper (`www.mgchemicals.com`)
- Dremel tool and wooden skewer as a drill bit
  - Doesn't damage the components underneath coating
  - Might remove the soldermask, but not a big deal...

Hardware Reverse Engineering

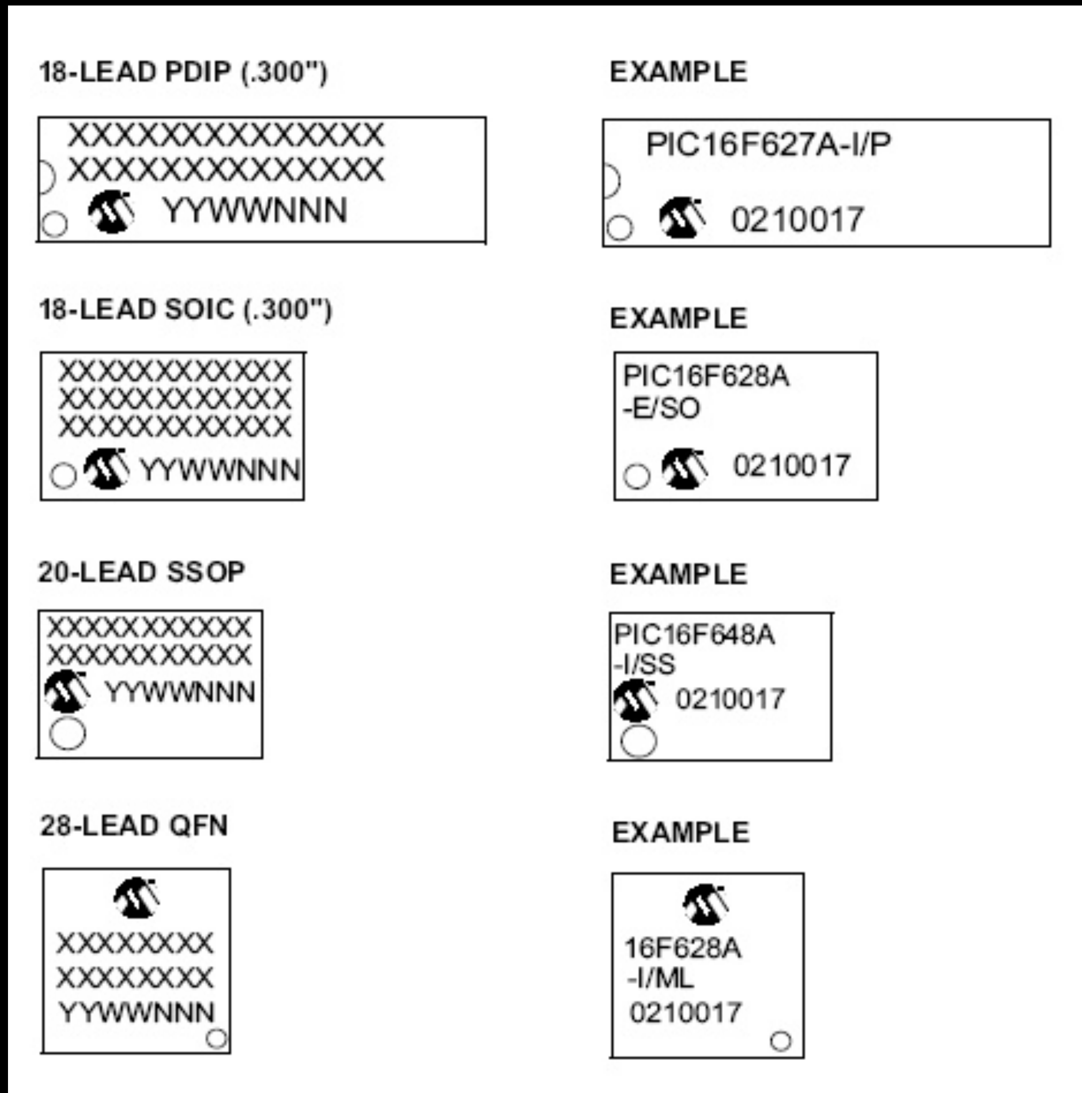
Component Identification

- Access to component will aid reverse engineering
- Most vendors and part numbers printed directly onto component (larger components)
- Surface mount or small devices use an abbreviated code
  - Not enough space on the package to print full information
  - Abbreviation details available in manufacturer data sheets
  - Educated/lucky guesses to help narrow down part

Component Identification 2

- Basic identification tips:
  - Look for manufacturer's logo
  - Look for alphanumeric string on part (if multiple text strings available, usually a manufacturing date code or speed rating)
  - Find data sheets (coming up next...)
- To help identify IC vendor logos: **`http://web.archive.org/web/20040210014748/http://www.elektronikforum.de/ic-id`**
- To help identify SMD markings: **`http://tinyurl.com/muy4qa`**

Component Identification 3



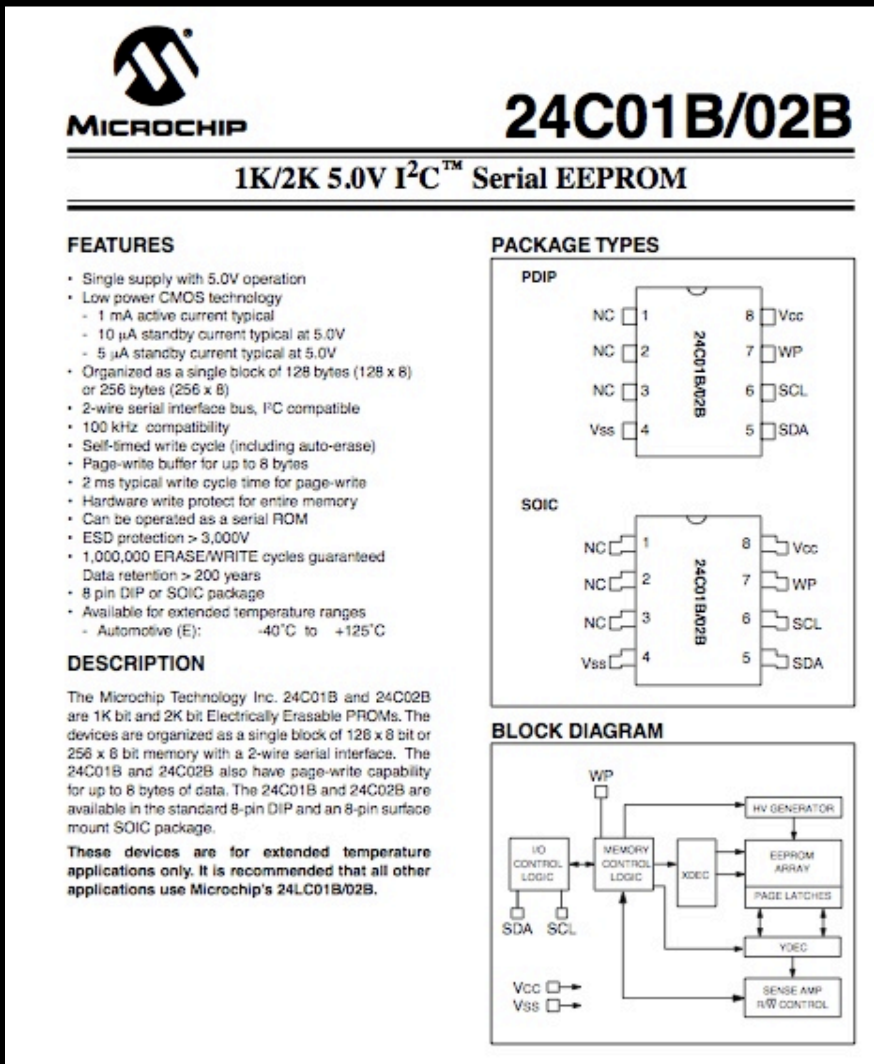
Component Identification 4

- Sometimes, sensitive targeted components are made intentionally difficult to access
- Some vendors remove identifiers and markings from ICs
  - Ex.: Stainless steel brush, small sander, micro-bead blast, laser etcher, or third-party
- May still be able to identify parts without the markings by probing or following important looking traces/signals

Finding Data Sheets

- Data sheets contain extremely useful technical component information:
  - Product overview
  - Pinout and pin function
  - Electrical parameters and functional limits
  - Application data
  - Package drawings

Finding Data Sheets 2



MICROCHIP

24C01B/02B

1K/2K 5.0V I²C™ Serial EEPROM

FEATURES

- Single supply with 5.0V operation
- Low power CMOS technology
  - 1 mA active current typical
  - 10 µA standby current typical at 5.0V
  - 5 µA standby current typical at 5.0V
- Organized as a single block of 128 bytes (128 x 8) or 256 bytes (256 x 8)
- 2-wire serial interface bus, I²C compatible
- 100 kHz compatibility
- Self-timed write cycle (including auto-erase)
- Page-write buffer for up to 8 bytes
- 2 ms typical write cycle time for page-write
- Hardware write protect for entire memory
- Can be operated as a serial ROM
- ESD protection > 3,000V
- 1,000,000 ERASE/WRITE cycles guaranteed
  Data retention > 200 years
- 8 pin DIP or SOIC package
- Available for extended temperature ranges
  - Automotive (E): -40°C to +125°C

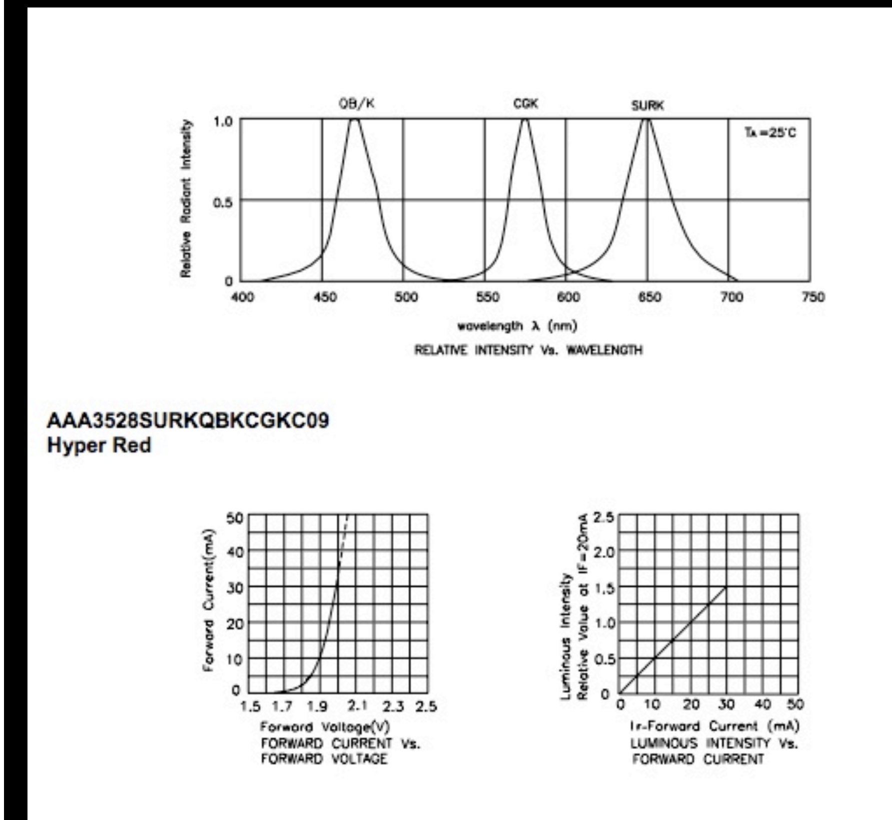
DESCRIPTION

The Microchip Technology Inc. 24C01B and 24C02B are 1K bit and 2K bit Electrically Erasable PROMs. The devices are organized as a single block of 128 x 8 bit or 256 x 8 bit memory with a 2-wire serial interface. The 24C01B and 24C02B also have page-write capability for up to 8 bytes of data. The 24C01B and 24C02B are available in the standard 8-pin DIP and an 8-pin surface mount SOIC package.

These devices are for extended temperature applications only. It is recommended that all other applications use Microchip's 24LC01B/02B.

PACKAGE TYPES

BLOCK DIAGRAM



Finding Data Sheets 3

- Many free and pay-for-search data sheet locator services online
  - Google, duh.
  - Octopart, `www.octopart.com`
  - Find Chips, `www.findchips.com`
  - Data Sheet Locator, `www.datasheetlocator.com`
  - IC Master, `www.icmaster.com`
  - PartMiner, `www.partminer.com`
  - ChipDB, `www.msarnoff.org/chipdb/`

Design-for-Manufacturability

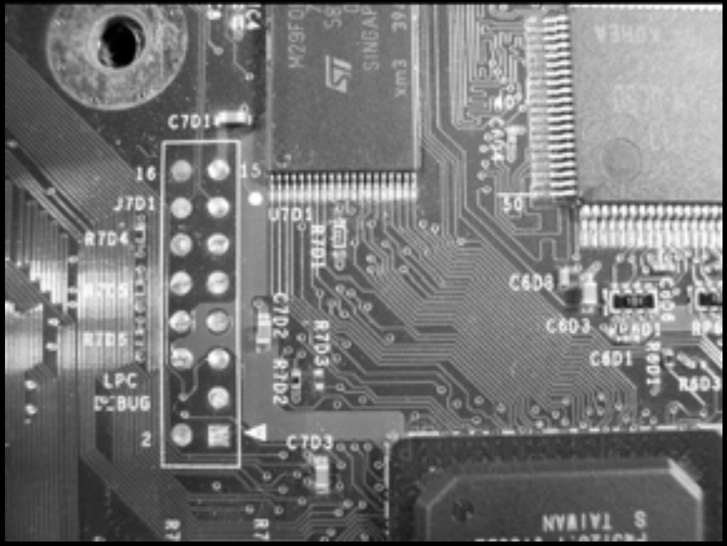
- Generally, manufacturers desire:
  - Full visibility into the system/circuit state
  - Unhindered access to key signals
  - Visual inspection capabilities
- Helps to keep rework/repair costs low, yield high, and failure analysis simple
  - It also helps hackers!

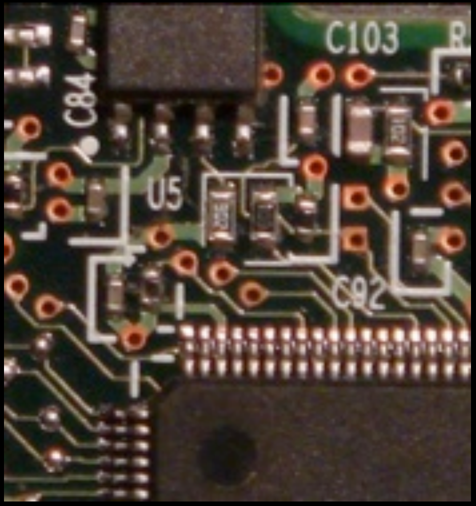
Design-for-Test

- Design in test structures that enable quick diagnostics of a system or circuit
  - Easy-to-access test/probe points
  - Industry-standard test interfaces
  - Proprietary test/debug ports (Ex.: Microchip PIC ICD2, Freescale BDM, Texas Instruments Spy-by-Wire)

Design-for-Test: Probe Points

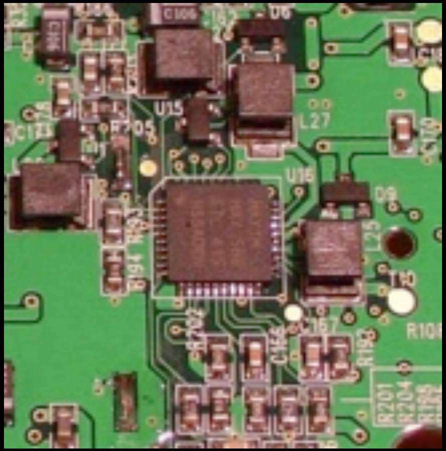
- Small circles are probe/test points
  - Indication of "interesting" signals - what's good for the engineer is good for the hacker
  - Sometimes easily identifiable by silkscreen outline or easy-to-access locations

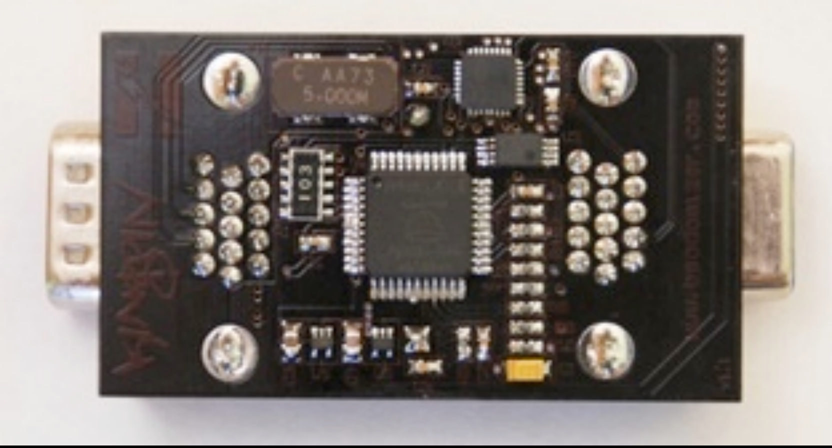




Probing Boards and Tracing Signals

- Look for test points and exposed traces/bus lines
- "Follow the copper"





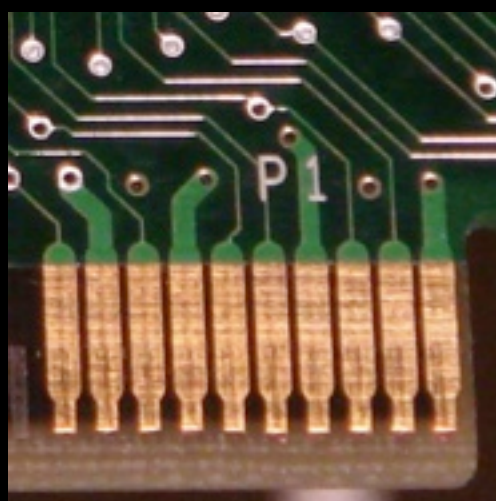
- For traces that are not exposed, use a continuity meter and "sweep" the board to find the connection

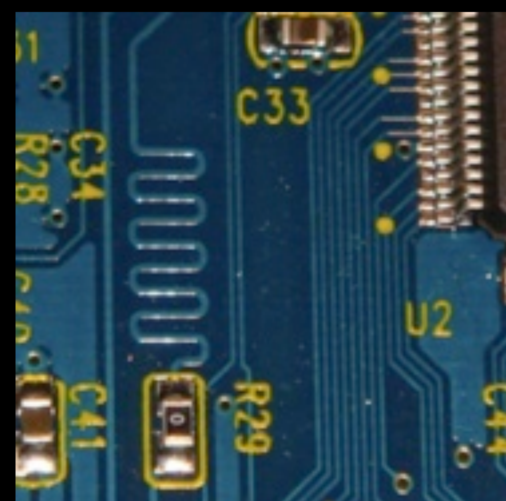
Probing Boards: Things to Look For

- Data being transferred across exposed and/or accessible address, data, and control buses
- Confusing trace paths to prevent easy reverse engineering
  - Hidden critical traces on inner board layers
- Use of buried vias
  - Connects between two or more inner layers but no outer layer
  - Cannot be seen from either side of the board
  - Increased manufacturing cost

Probing Boards: Layout Motifs

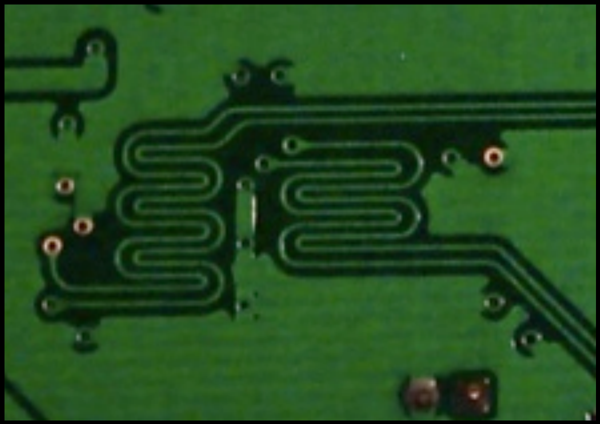
- Determining traces is a time consuming process
- Rely on heuristics to identify trace function:
  - Power traces are thick, usually short
  - Impedance controlled signals usually thick and long
    - Often clock, high-speed data, or other critical line

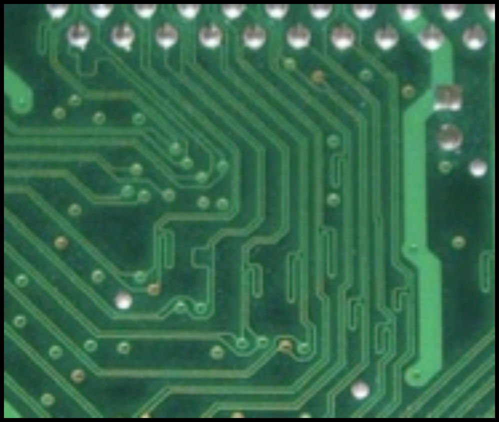


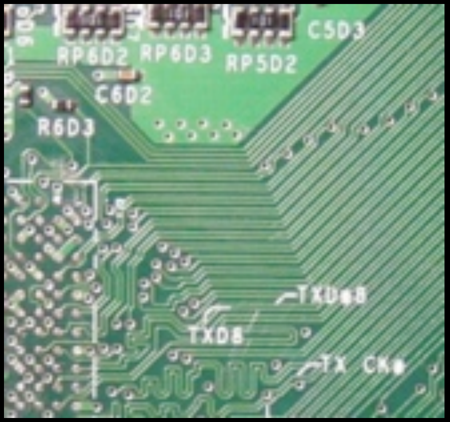


Probing Boards: Layout Motifs 2

- Pairs of traces indicate differential signaling
- "Zig-zag" traces indicate length-matched busses (typically high-speed)
- Traces of similar function are grouped together







Memory and Programmable Logic

Memory and Programmable Logic

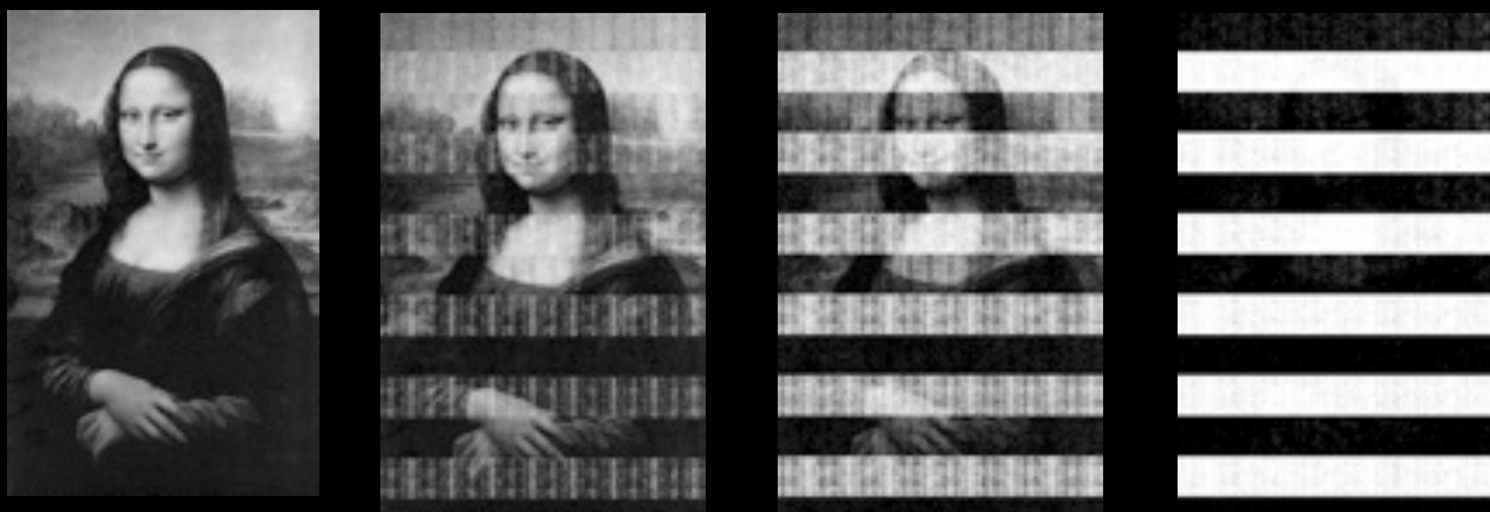
- Most memory is notoriously insecure
  - Not designed with security in mind
  - Can read most memory with an off-the-shelf, general purpose device programmer
  - Serial EEPROMs can usually be read in-circuit
    - Ex.: India Electronic Voting Machines, April 2010, `http://indiaevm.org/`
- SRAM-based FPGAs vulnerable to attack
  - Must load configuration from external memory
  - Bit stream can be monitored to retrieve entire configuration
  - Bit stream may be encrypted

Memory and Programmable Logic 2

- Remnants may exist and be retrievable from devices long after power is removed
  - Could be useful to obtain program code, temporary data, crypto keys, etc.
  - "Data Remanence in Semiconductor Devices," **`www.usenix.org/publications/library/proceedings/sec01/gutmann.html`**
  - Ex.: "An Integrated Approach to Recovering Deleted Files from NAND Flash Data," **`www.ssddfj.org/papers/SSDDFJ_V2_1_Luck_Stokes.pdf`**

Memory and Programmable Logic 3

- Ex.: Cold Boot attacks, `http://citp.princeton.edu/memory/`



5 Seconds | 30 Seconds | 60 Seconds | 5 Minutes

Memory and Programmable Logic 4

- Security fuses and boot-block protection
  - Enabled for "write-once" access to a memory area or to prevent full read back
  - May be bypassed with die analysis attacks or electrical faults
    - "Design Principles for Tamper-Resistant Smartcard Processors," **`www.cl.cam.ac.uk/~mgk25/sc99-tamper.pdf`**
    - "Copy Protection in Modern Microcontrollers," **`www.cl.cam.ac.uk/~sps32/mcu_lock.html`**

Memory and Programmable Logic 5

- Once firmware/data is retrieved, can reverse engineer using standard software techniques
- Disassemble, modify, recompile, and reprogram device, if desired
- Ex.: IDA Pro, `www.hex-rays.com`
- Ex.: PICDisasm, `http://hagi-online.org/picmicro/picdisasm_en.html`

External Interfaces

External Interfaces

- Usually a product's lifeline to the outside world
  - Manufacturing tests, field programming/upgrading/ debugging, peripheral connections
  - Ex.: RS232, USB, Firewire, Ethernet
  - Proprietary test/debug ports (Ex.: Microchip PIC ICD2, Freescale BDM, Texas Instruments Spy-by-Wire, Nokia F-Bus/M-Bus)
- Wireless interfaces also at risk
  - Ex.: 802.11, Bluetooth, ZigBee, ANT+
- Any interface that connects to a third-party may contain information that is useful for an attack
  - Could possibly obtain data, secrets, etc.

External Interfaces 2

- Look for obfuscated interfaces
  - Ex.: Proprietary or out-of-the-ordinary connector types, hidden access doors or holes, underneath battery holders
  - Many times, test points just hidden by a sticker

External Interfaces 3

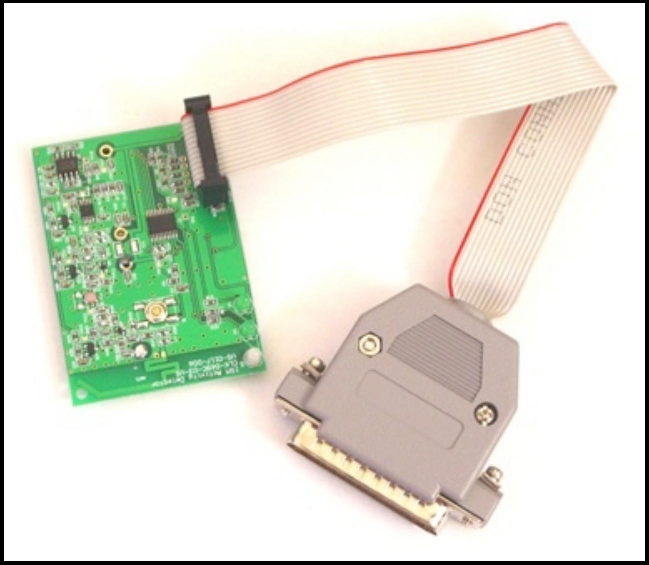
- Use multimeter or oscilloscope to probe and determine functionality
  - Logic state of pins can help with an educated guess
  - Ex.: Pull pins high or low, observe results, repeat
- Monitor communications using H/W or S/W-based protocol analyzer
  - Ex.: Bus Pirate, **`www.buspirate.com`**
  - RS232 and parallel port: PortMon
  - USB: SnoopyPro, SourceUSB
- Send intentionally malformed/bad packets to cause a fault
  - If firmware doesn't handle this right, device could trigger unintended operation useful for an attack

JTAG

- JTAG (IEEE 1149.1, Joint Test Action Group) interface is often the Achilles' heel
- Industry-standard interface for testing and debugging
  - Ex.: System-level testing, serial boundary-scanning, low-level testing of dies and components, firmware debugging (single stepping and setting breakpoints)
  - **`http://en.wikipedia.org/wiki/Joint_Test_Action_Group`**
- Can provide a direct interface to hardware
  - Ex.: Flash memory reprogramming

JTAG 2

- Five connections (4 required, 1 optional):
  - ← TDO = Data Out (from target device)
  - → TDI = Data In (to target device)
  - → TMS = Test Mode Select
  - → TCK = Test Clock
  - → /TRST = Test Reset (optional)
- Typical JTAG header pinouts:
  `www.jtagtest.com/pinouts/`



JTAG 3

- Many low-cost JTAG interfaces available (usually device/tool-specific)
  - Ex.: `www.sparkfun.com/commerce/advanced_search_result.php?keywords=jtag`
- Old school parallel port interfaces can be built for only a few dollars
  - Ex.: `http://jtag-arm9.sourceforge.net/circuit.txt`
  - Ex.: `ftp://www.keith-koep.com/pub/arm-tools/jtag/jtag05_sch.pdf`

JTAG 4

- Many development environments provide support for JTAG interfaces
  - Low-level functionality is abstracted from the user
- Some open-source S/W tools exist
  - Ex.: Open On-Chip Debugger (OpenOCD), **`http://openocd.berlios.de/web/`**
  - Ex.: UrJTAG (Universal JTAG Library), **`www.urjtag.org`**

JTAG 5

- Removing JTAG functionality from a device is difficult
  - Designers usually obfuscate traces, cut traces, or blow fuses, all of which can be repaired by an attacker
  - Inconvenient, because it would remove programming/debug/testing capabilities for the legitimate users!
  - May be password protected
- Ex.: Barnaby Jack's Vector Rewrite Attack,

  `www.securityfocus.com/columnists/446` and **`www.juniper.net/solutions/literature/white_papers/Vector-Rewrite-Attack.pdf`**

Advanced Techniques

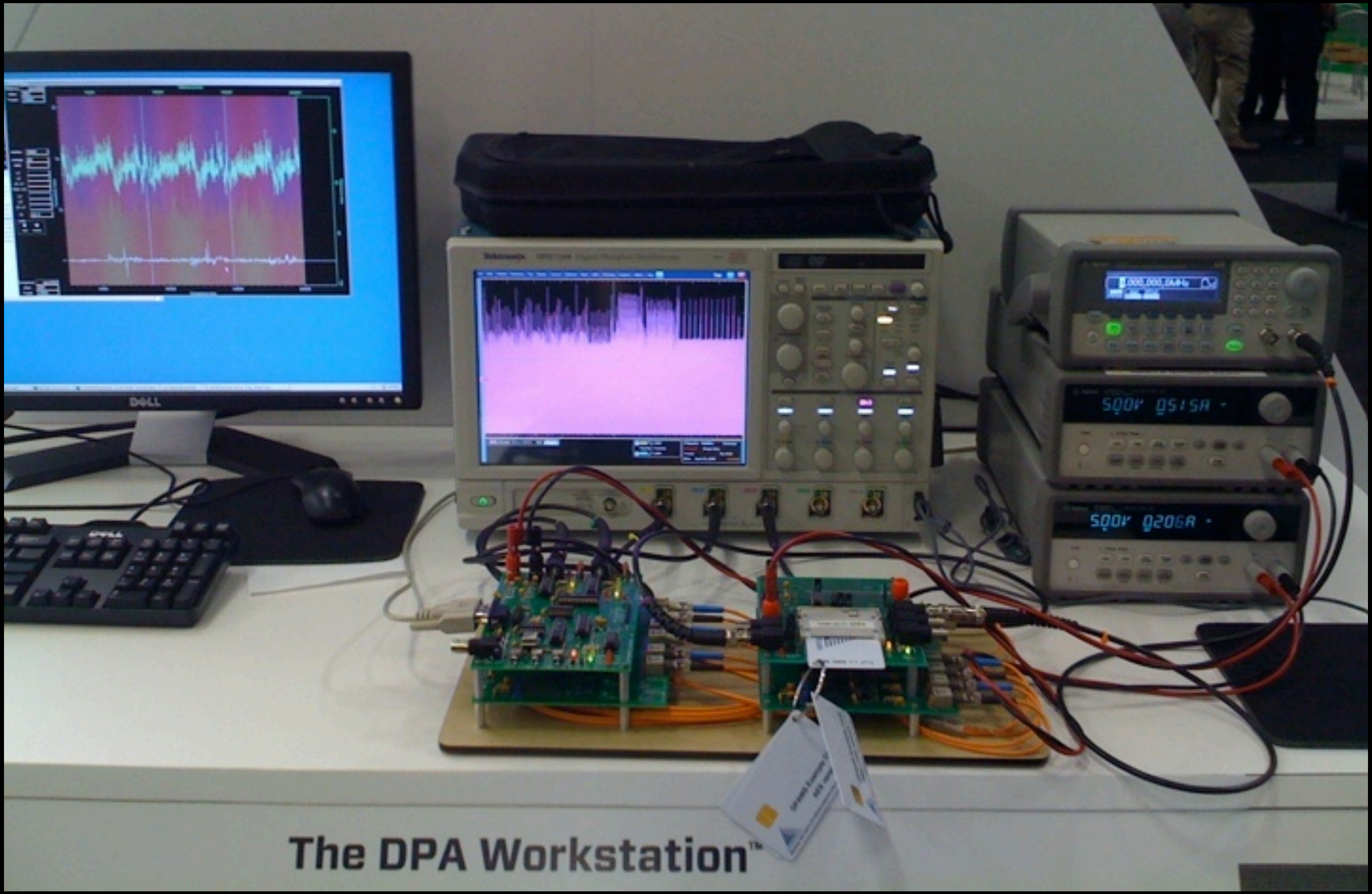
Side-Channel Attacks

- All devices leak information
  - Time
  - Power consumption
  - EMI (electromagnetic interference)
    - "Electromagnetic Radiation from Video Display Units," **www.jya.com/emr.pdf**
    - "The EM Side–Channel(s): Attacks and Assessment Methodologies," **www.research.ibm.com/intsec/emf-paper.ps**
  - Light and Sound
    - "Information Leakage from Optical Emanations," **www.applied-math.org/optical_tempest.pdf**
    - "Optical Time-Domain Eavesdropping Risks of CRT Displays," **www.cl.cam.ac.uk/~mgk25/ieee02-optical.pdf**

Side-Channel Attacks 2

- Can be used to retrieve secrets (keys, PIN) or reverse engineer firmware (program flow, crypto)
- Ex.: Side Channel Cryptanalysis Lounge, **`www.crypto.ruhr-uni-bochum.de/en_sclounge.html`**
- Ex.: "Side Channel Analysis on Embedded Systems," Job de Haas, HITB 2009, **`http://conference.hackinthebox.org/hitbsecconf2009kl/materials/D2T1%20-%20Job%20De%20Haas%20-%20Side%20Channel%20Analysis.pdf`**

Side-Channel Attacks 3



Side-Channel Attacks: Power Analysis

- Unintended physical leakage of information based on power consumption
- Simple Power Analysis (SPA)
  - Attacker directly observes power consumption
  - Varies based on microprocessor operation
  - Easy to identify intensive functions (cryptographic)
- Differential Power Analysis (DPA)
  - Statistical methods to determine secret information on a device
  - Pioneered by Cryptography Research, Inc.

Side-Channel Attacks: Power Analysis 2

- "Overview of Differential Power Analysis," `www.cryptography.com/resources/whitepapers/DPA.html`
- "Power Analysis Attacks - Revealing the Secrets of Smartcards," `www.dpabook.org`
- OpenSCA - A Matlab-based open source framework for side-channel attacks, `http://opensca.sourceforge.net`

Side-Channel Attacks: Clock and Timing

- Attacks rely on changing or measuring timing characteristics of the system
- Active (Invasive) timing attacks
  - Vary clock (speed up or slow down) to induce failure or unintended operation
- Passive timing attacks
  - Non-invasive measurements of computation time
  - Different tasks take different amounts of time

Silicon Die Analysis

- Extremely useful depending on attack goals
  - Simple imaging to gather clues
  - Key/algorithm extraction from ICs
  - Retrieve contents of Flash, ROM, FPGAs, other non-volatile devices
  - Cutting or repairing silicon structures (security fuses, traces, etc.)
- Like reversing circuitry, but at a microscopic level

Silicon Die Analysis: IC Decapsulation

- Decapsulation tools used to "delid" or "decap" the top of the IC housing
- Uses chemical or mechanical means (or both)
- Will keep the silicon die intact while removing the outer material
  - Depending on the decapping method used, the product will still function!

Silicon Die Analysis: IC Decapsulation 2

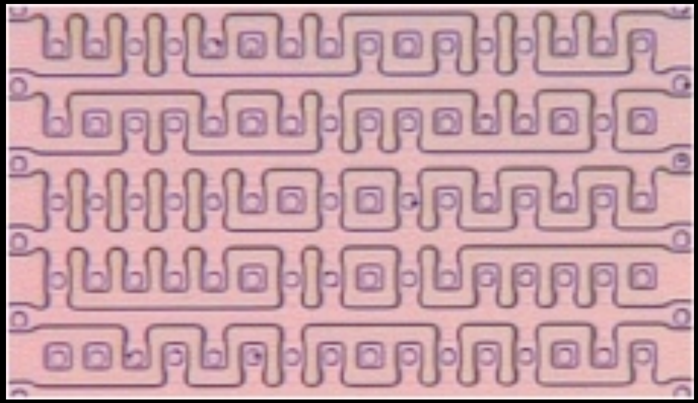
- Tools: Nippon Scientific (**`www.nscnet.co.jp/e`**), Nisene Technology Group (**`www.nisene.com`**), ULTRA TEC Manufacturing (**`www.ultratecusa.com`**), approx. $30k new, $15k used
- Services:
  - Flylogic, **`www.flylogic.net`**
  - MEFAS, Inc. (Micro Electronics Failure Analysis Services), **`www.mefas.com`**, approx. $50 and 2-day wait for a single device

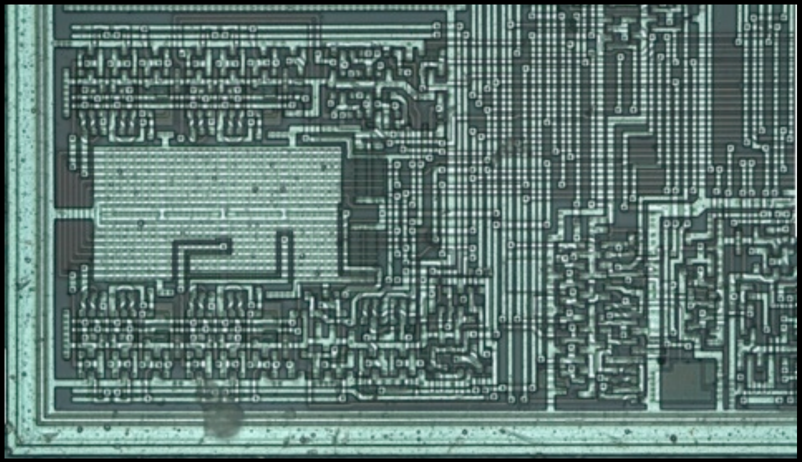
Silicon Die Analysis: Scanning Electron Microscope

- Used to perform chip-/gate-level inspection of the physical die
- Images can be used to:
  - Determine manufacturer/chip type for hacking or competitive analysis
  - Determine attack vectors
  - Reverse engineer chip functionality

Silicon Die Analysis: Scanning Electron Microscope 2

- Will usually need to remove metal or other layers before getting access to gate structures
- Depending on ROM size and properties, can visually recreate contents



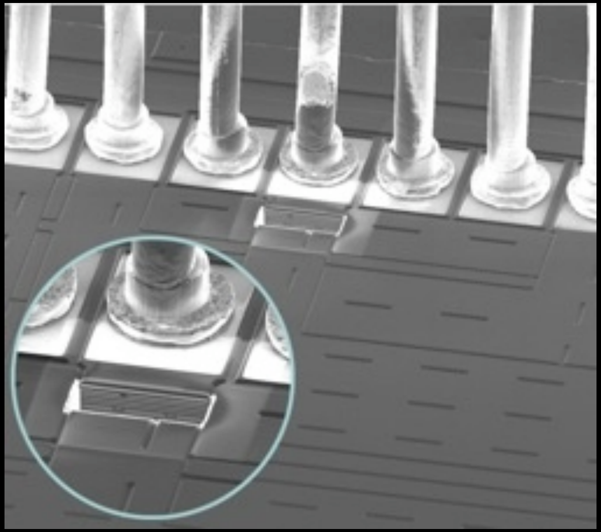


Silicon Die Analysis: FIB (Focused Ion Beam)

- Send a focused stream of ions onto surface of the chip
  - Beam current/velocity and optional use of gas/vapor changes the function
- Imaging
- Cutting
  - Ex.: Cut a bond pad or trace from the die
- Deposition
  - Ions react with a chemical vapor to precipitate a metal film
  - Ex.: Add a jumper/reconnect a trace on the die

Silicon Die Analysis: FIB (Focused Ion Beam) 2

- Ex.: `www.fei.com/products/focused-ion-beams/`
- Ex.: Fibics Incorporated, `www.fibics.com`
- Ex.: FIB International, `www.fibinternational.com`



Picture: Fibics Incorporated

Silicon Die Analysis 2

- "Real" equipment still fairly expensive, but can find in academic environment, get from surplus, or go low-tech:
  - Fuming Nitric Acid (HNO3)
  - Acetone
  - Microscope
  - Micropositioner w/ sewing needle

Silicon Die Analysis 3

- Required reading/viewing:
  - Chris Tarnovsky/Flylogic Engineering's Analytical Blog, `www.flylogic.net/blog`
  - "Hack a Sat-TV Smart Card," `www.wired.com/video/hack-a-sattv-smart-card/1813637610`
  - "Hacking Silicon: Secrets from Behind the Epoxy Curtain," Bunnie Huang, ToorCon 7, `www.toorcon. org/2005/slides/bunnie-hackingsilicon.pdf`
  - "Hardware Reverse Engineering," Karsten Nohl, 25C3, `http://tinyurl.com/ya3s56r`
  - "Deep Silicon Analysis," Karsten Nohl, HAR 2009, `har2009.org/program/events/149.en.html`

Example

(There are many to choose from...)

Smart Parking Meters

- Parking industry generates $28 billion annually worldwide
- Where there's money, there's risk for fraud and abuse
- Attacks/breaches can have serious fiscal, legal, and social implications
- Collaboration w/ Jake Appelbaum and Chris Tarnovsky
- Released @ BH USA 2009
- Full details at `www.grandideastudio.com/portfolio/smart-parking-meters/`

San Francisco MTA

- Part of a $35 million pilot program to replace 23,000 mechanical meters with "smart" parking meters in 2003
- Infrastructure currently comprised of MacKay Guardian XLE meters
- Stored value smart card
  - $20 or $50 quantities
  - Can purchase online w/ credit card or in cash from selected locations
  - Dispose when value runs out

San Francisco MTA 2

- Easy to replay transaction w/ modified data to obtain unlimited parking
  - Determined solely by looking at oscilloscope captures of smartcard communications
  - Succeeded in three days

Information Gathering

- A chance encounter w/ Department of Parking & Transportation technician on the streets of SF
  - Ask smart, but technically awkward questions to elicit corrections
- Crawling the Internet for specific information
  - Product specifications, design documents, etc.
- How It's Made, Season 5, Episode 7: `www.youtube.com/watch?v=1jzEcblRLEI`

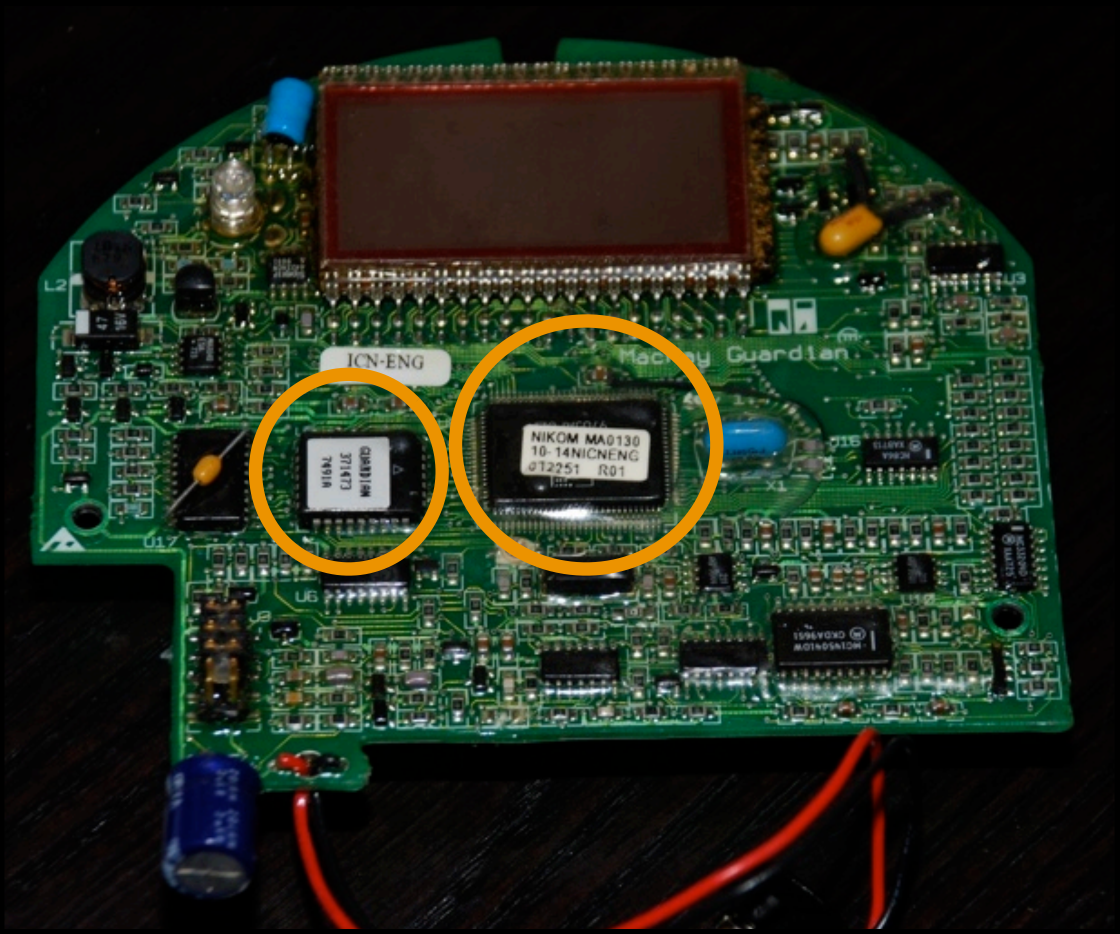
Information Gathering 2

```
# From: xxx <xxx at jjmackay dot ca>
# Date: Wed, 14 Mar 2001 10:27:29 -0400


I am learning how to use CVS and as part of this process I set up a test
repository to 'play' with.



D:\src\working\epurse\cvstest>cygcheck -s -v -r -h

Cygnus Win95/NT Configuration Diagnostics
Current System Time: Wed Mar 14 09:39:50 2001

Win9X Ver 4.10 build 67766446  A

Path:   /cygdrive/c/NOVELL/CLIENT32
        /cygdrive/c/WINDOWS
        /cygdrive/c/WINDOWS/COMMAND
        /usr/bin
        /cygdrive/c/JJMACKAY/MET_TALK
        /cygdrive/c/JJMACKAY/UTILITY



GEMPLUS_LIB_PATH = `C:\WINDOWS\GEMPLUS'


Found: C:\cygwin\bin\gcc.exe
Found: C:\cygwin\bin\gdb.exe


xxx, Sr. Software Designer
```

http://www.cygwin.com/ml/cygwin/2001-03/msg00842.html

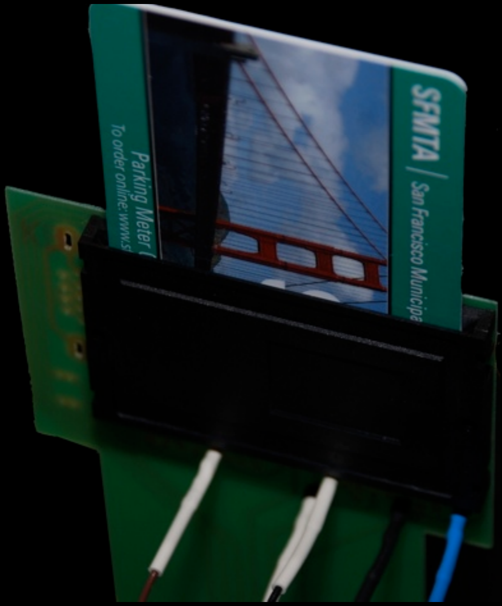
Meter Disassembly: MacKay Guardian

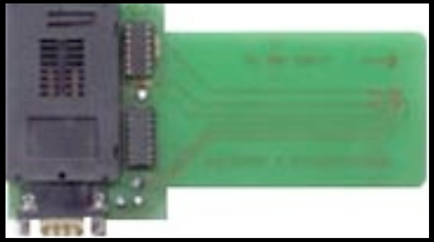
Meter Disassembly: MacKay Guardian 2



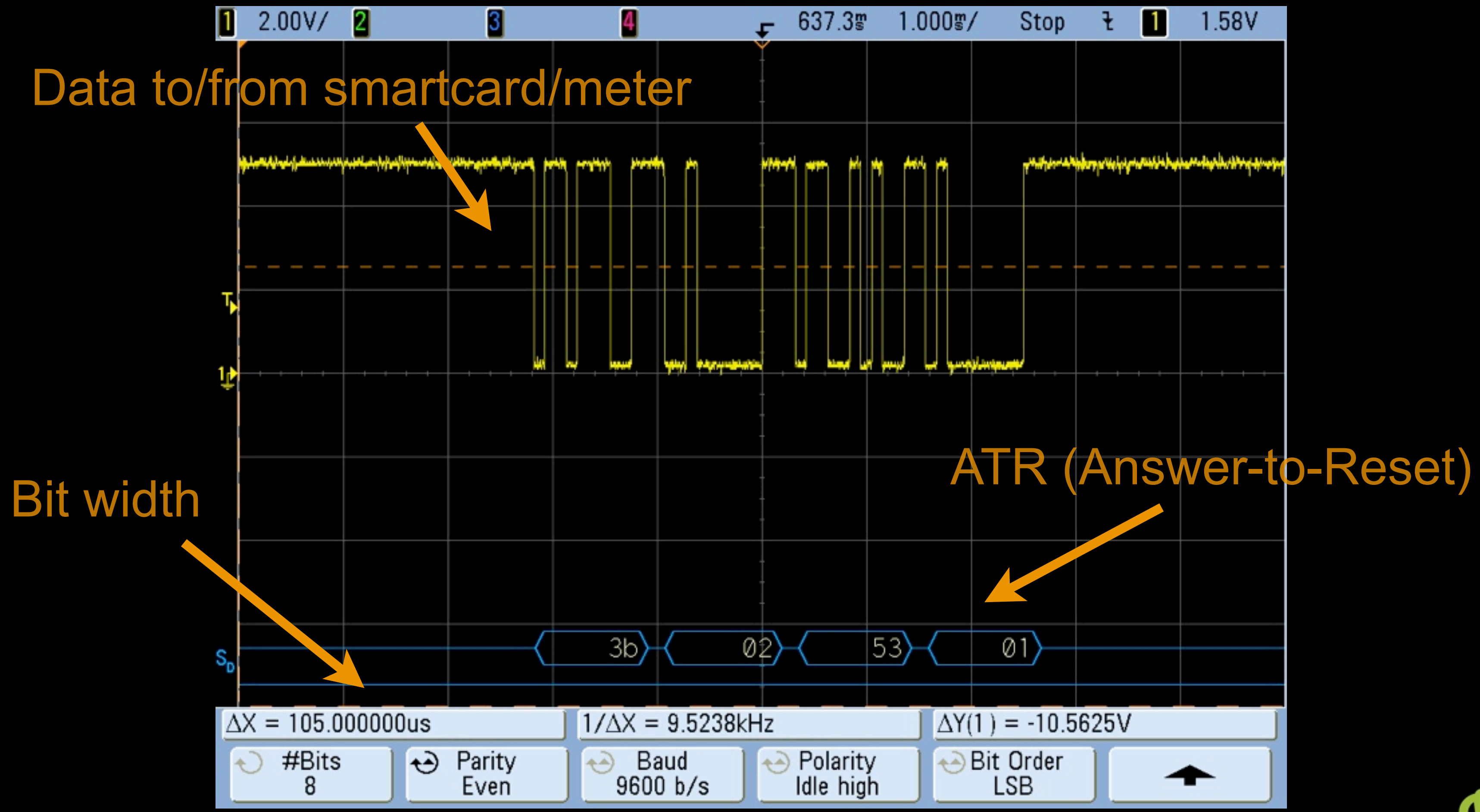
Smartcard Communications Monitoring

- Used "shim" between smartcard and meter
  - Unpopulated Season 2 Interface
- Monitored I/O transaction w/ digital oscilloscope
- Asynchronous serial data @ 9600, 8E1 captured and decoded
  - Correct baud rate determined by measuring bit width on scope





Smartcard Communications Monitoring 2



Smartcard Protocol Decoding

- Captured multiple transactions to gather clues on operation
  - Different valued cards
  - Different serial numbers
- Based on what values changed per transaction & per card, could narrow down what data meant what
- Decoded transaction functionality by hand, no computer needed!

Deduction of a Single Unit ($0.25)

Meter	*Card*
	[4 byte responses unless noted]
Update Balance 1 Current Value A1 →	
←	OK (2)
Update Balance 1 Current Value A2 →	
←	OK (2)

- By updating the Balance 1 Value (8 bytes), CTC1 automatically increments
- CTC1 is the only value that changes on the card during the entire transaction!

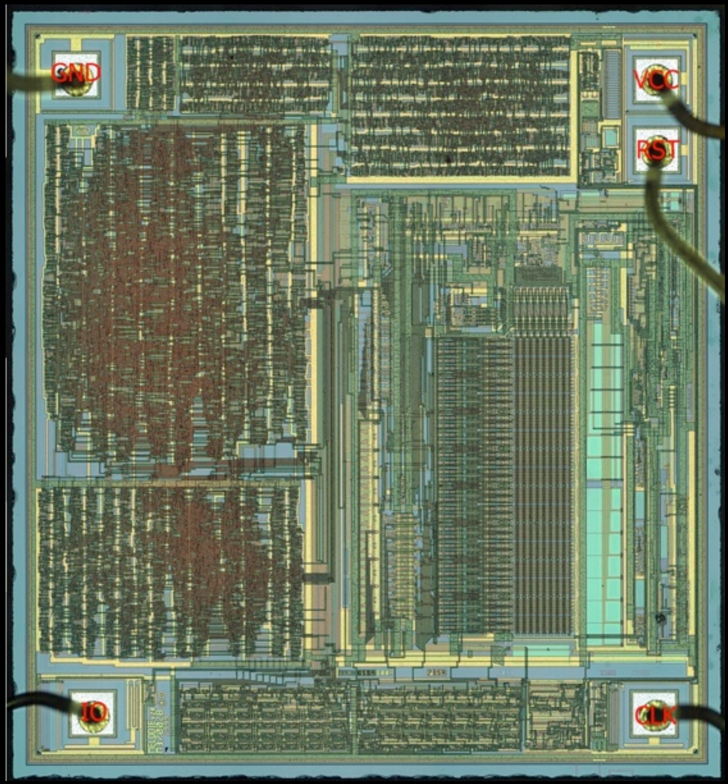
Computation of Card Value

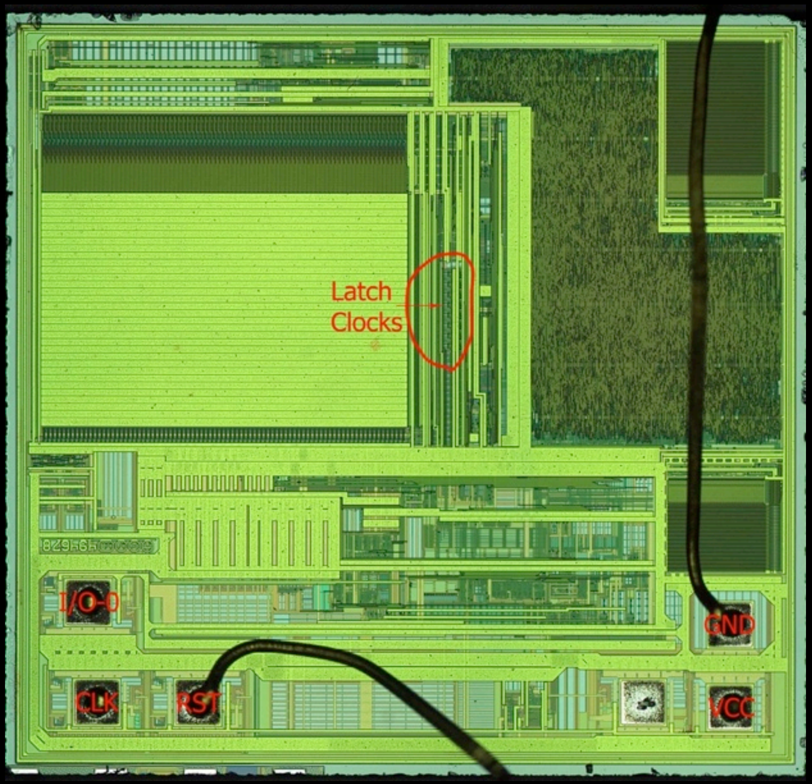
- Maximum card value = (Balance 2 - 95d)
  - Ex.: 0x0AF (175d) - 95d = 80 units
    - 80 * $0.25 = $20
  - Ex.: 0x127 (295d) - 95d = 200 units
    - 200 * $0.25 = $50

Smartcard Die Analysis

- Purchased and decapsulated multiple cards to look for clues of manufacturer and functionality
- Visually identified that two different smartcard types exist
  - Gemplus GemClub-Memo (ASIC)
  - 8051 microcontroller *emulating* GemClub-Memo
- Dependent on card serial number
  - Older cards are ASIC, newer cards are MCU
- Microcontroller has potential for hidden/ undocumented commands
  - One could retrieve the code from the card and reverse engineer (we didn't)

Smartcard Die Analysis 2





Protocol Emulation

- First attempt to replay exact transaction captured w/ scope
  - Microchip PIC16F648A
  - Written in C using MPLAB + CCS PIC-C
  - Challenge for code to be fast enough and incorporate required short delays while still be readable/useful C

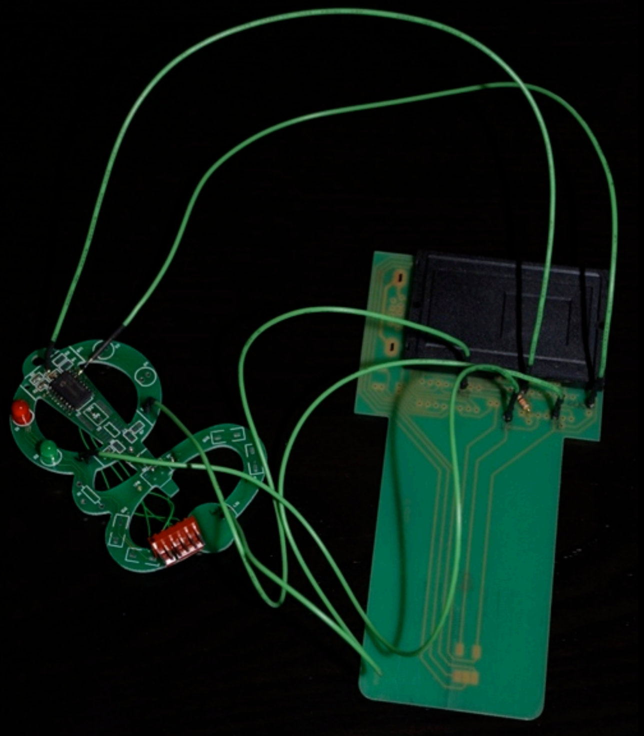
Protocol Emulation 2

- Then, modified code to change various values until success
  - Knowing how "remaining value" is computed, what happens if we change Balance 2 to 0xFFF?

Protocol Emulation 3

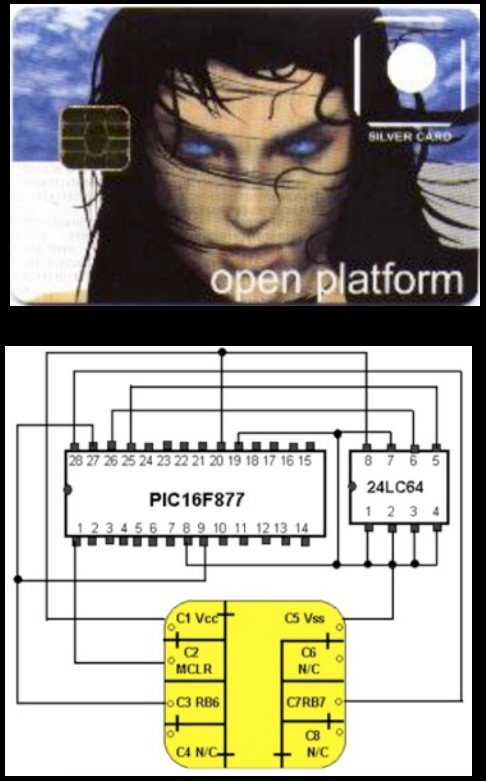
- As icing on the cake, ported code to Silver Card (PIC16F877-based smartcard)
  - PIC-based smartcards have been popular for satellite TV hackers for years, so required tools are readily available
  - Ex.: `http://interesting-devices.com`

Hardware Evolution



1) Custom PCB + shim

2) MM2 card w/ external PIC



3) Silver Card: PIC16F877 smartcard

San Francisco MTA Results

Final Thoughts

- Hardware is now more accessible to hackers than ever before
- The line is now blurred between HW & SW
- New skills and techniques continually being developed and shared
- Learn from history and other people's mistakes to...
  - Make your products better
  - Break someone else's products

Open Lab!