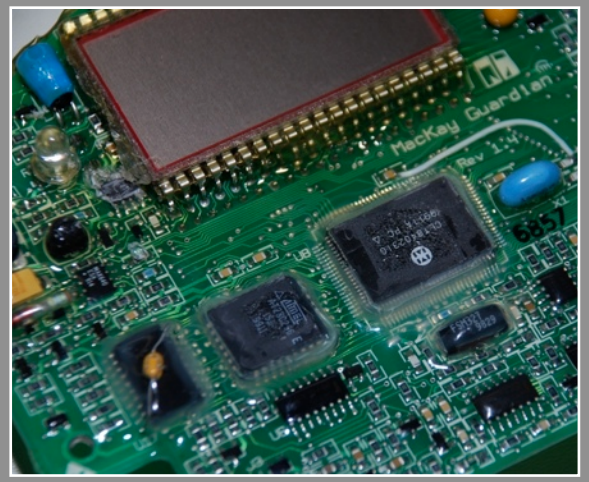
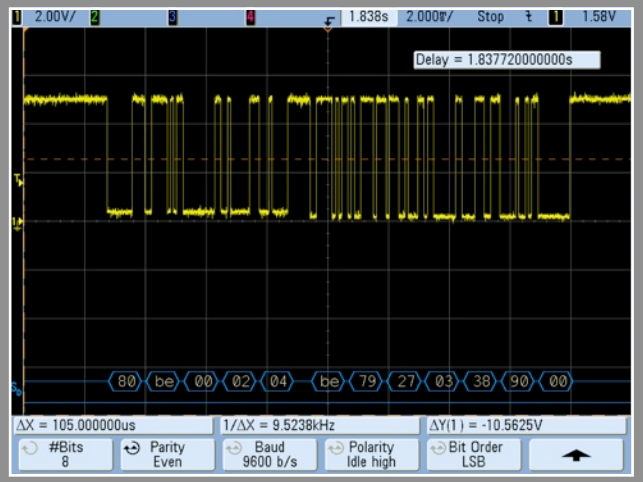
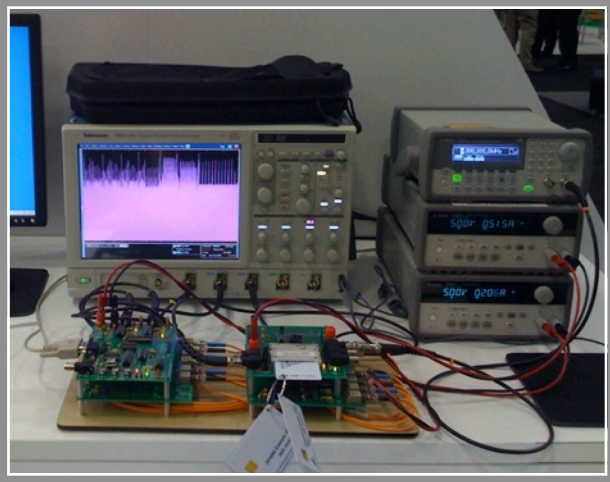
# Hacks and Attacks: Examples of Electronic Device Compromise



Joe Grand, Grand Idea Studio, Inc.

ESC-343

# The Plan

- A look into hardware hacking and why it's becoming so easy
- Show lots of examples
  - Learn from history and other people's mistakes to make your products better!

# 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 Attack Hardware? For Good?

- 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

# Why Attack Hardware? For Evil?

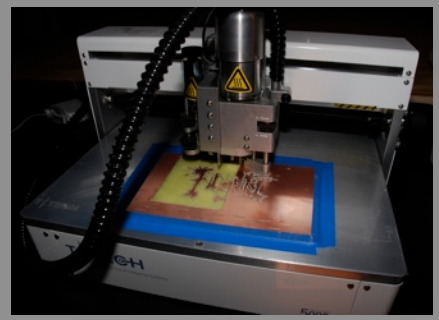
- Theft of service
  - Obtaining a service for free that normally costs $$$
- Competition/cloning
  - Specific theft of information/data/IP to gain a marketplace advantage
- User authentication/spoofing
  - Forging a user's identity to gain access to a system

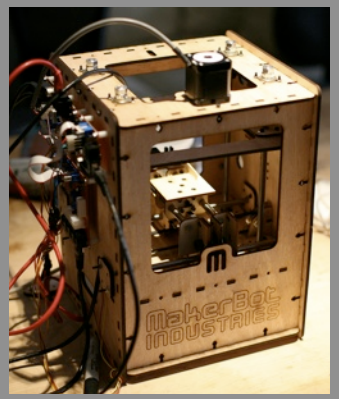
# Easy Access to Tools

- Cost can be less than setting up a SW dev. environment
- Pre-made, entry-level packages available
  - Soldering iron, solder/desolder accessories, multimeter, oscilloscope
- Equipment available for cheap from eBay/surplus markets
  - Scopes, logic analyzers, device programmers, spectrum analyzers, microscope, FIB/SEM
- Low-cost/open source design tools
  - Ex.: EAGLE, gEDA, PCB123

# Easy Access to Manufacturing

- PCB Fabrication and Assembly
  - Many production houses available online
  - Quick turn, low cost
- Rapid Prototyping
  - Laser cutter
  - CNC
  - PCB prototype machine
  - 3D printing
    - Open-source solutions
    - MakerBot, RepRap,  Fab@home



# Easy Access to Information

- Open source hardware and DIY sites becoming commonplace
- People are publishing their new work daily
  - Pictures, videos, source code, schematics, Gerber plots
- Hackaday, Instructables, Harkopen, Adafruit, Circuit Cellar, Nuts & Volts, MAKE, EDN, YouTube, etc.

# Easy Access to Other People

- You don't have to live in a bubble anymore (if you don't want to)
- Can outsource tasks to people with specific/ specialized skills
- Hackerspaces
  - Local venues for sharing equipment and resources
  - HackerspaceWiki, `http://hackerspaces.org`
- Workshops
  - Public, membership-based organizations (like a health club)
  - Classes and training available

# Easy Access to Other People 2

- Various Forums & Conferences
  - Black Hat, DEFCON, ToorCon, HOPE, ShmooCon, CCC, Hacking at Random, Hack in the Box, Embedded Systems Conference, etc.

# Hardware Hacking Methodology

# Hardware Hacking Methodology 2

- There's never only *one* correct process
- It's all about gathering clues
- Major subsystems:
  - Information gathering
  - Hardware teardown
  - External interface analysis
  - Silicon die analysis
  - Firmware reversing

# 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

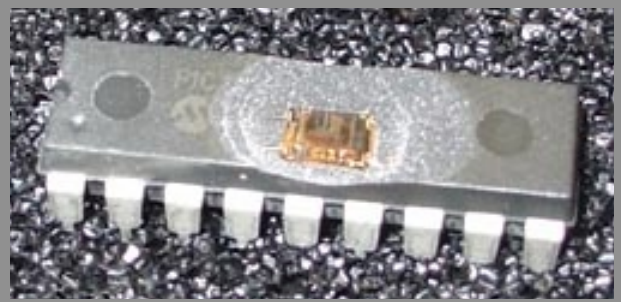
- Hardware and electronics disassembly and reverse engineering
- Get access to the circuitry
- Component and subsystem identification
- Gives clues about design techniques, potential attacks, and system functionality
- Typically there are similarities between older and newer designs
  - Even between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

# External Interface Analysis

- Communications monitoring
- Protocol decoding and/or emulation
- 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.
  - Ex.: Bus Pirate, `http://buspirate.com`

# Silicon Die Analysis

- Supremely 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 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

Wired.com, Hack a Sat-TV Smart Card

# Silicon Die Analysis 3

- Required reading/viewing:
  - "Hack a Sat-TV Smart Card," `www.wired.com/video/hack-a-sattv-smart-card/1813637610`
  - Chris Tarnovsky/Flylogic Engineering's Analytical Blog, `www.flylogic.net/blog`
  - "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`

# 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
- Quick run through w/ *strings* and hex editor to pick most interesting area to begin with
- Gives clues to possible entry/access points to administrative menus or ideas of further attacks

# Firmware Reversing 2

- Disassembly and reverse engineering using IDA, etc.
- Modify, recompile, and reprogram device, if desired
- 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

# 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

# Attack Examples

- e-Voting
- Authentication tokens
  - iButton DS1991
- Microcontrollers
  - Microchip PIC Configuration Fuses
  - MSP430 Serial Bootstrap Loader
- Fare collection
  - Boston MBTA CharlieCard/CharlieTicket
  - San Francisco Smart Parking Meter

# e-Voting Machines

- Massive security problems with devices around the world
- Casting multiple votes, tampering with election configurations and data, easily changing firmware, remote detection of voting via TEMPEST monitoring
- Ex.: `www.eff.org/issues/e-voting/`
- Ex.: `www.avirubin.com/vote/`
- Ex.: `http://wijvertrouwenstemcomputersniet.nl/English`

# iButton

- www.maxim-ic.com/products/ibutton/
- Rugged, portable data container
  - Ex.: Memory, RTC, data logging, ID
- Unique 64-bit ID (non-secret) for each device
- 1-wire Interface
  - Actually, 2 wires (clock/data and ground)
  - Parasitically-powered
  - 16kbps (standard) and 142kbps (overdrive)
- Easy to emulate/clone transaction
  - Ex.: www.reteam.org/board/showthread.php?t=1332

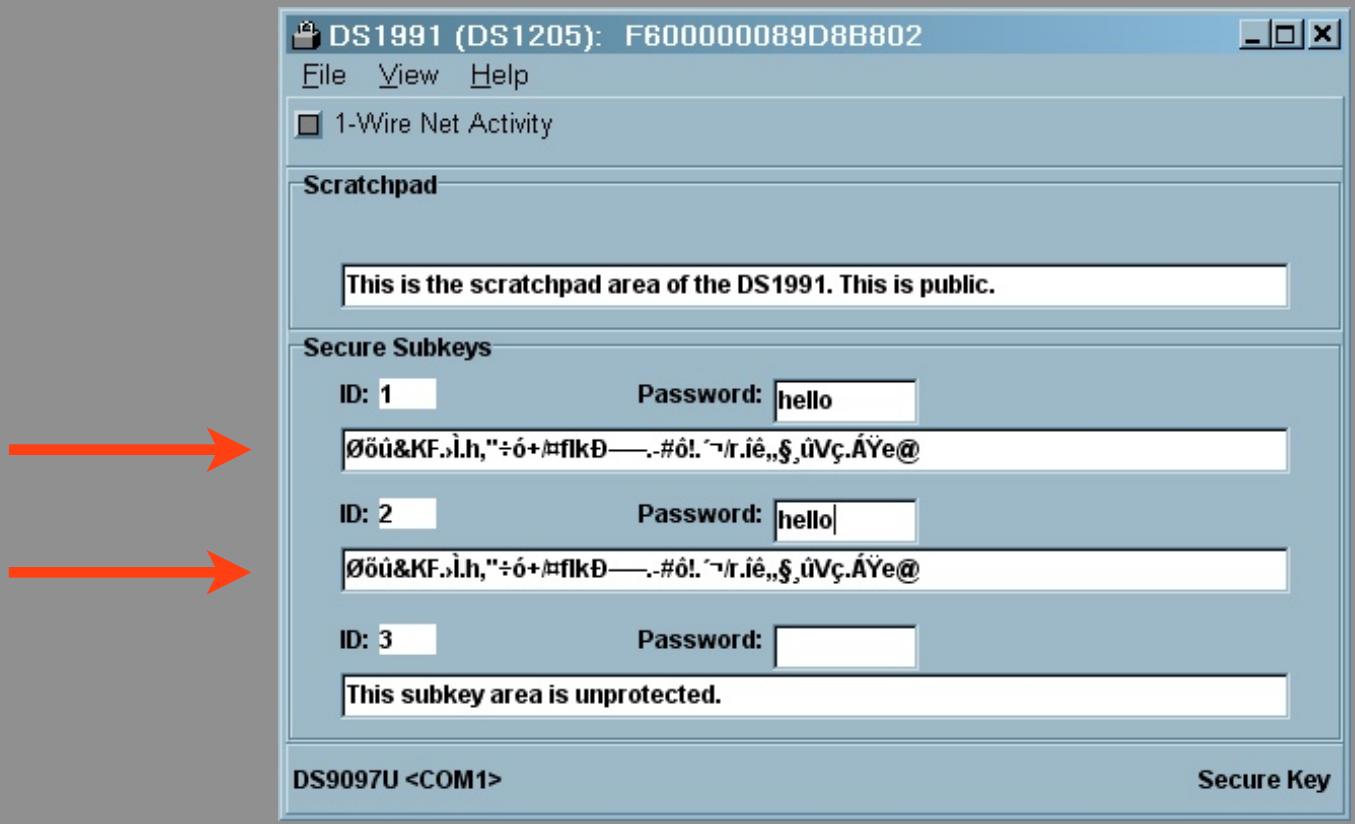


# iButton DS1991

- Used for cashless transactions (e.g., parking meters, public transportation) & access control
- 1,152 bits of non-volatile memory split into three 384-bit (48-byte) containers known as “subkeys”
- Each subkey is protected by an independent 8-byte password
- Only the correct password will grant access to the data stored within each subkey area and return the 48-bytes
  - Incorrect password supposed to return 48-bytes of "random" data

# iButton DS1991 2

- Initial experiments with iButton Viewer (part of free iButton-TMEX SDK) showed that "random" response is actually based on input password



# iButton DS1991 3

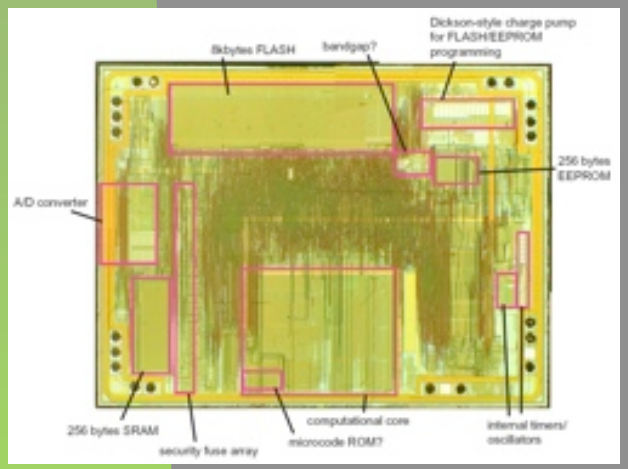
- For any given character (256 possibilities), a unique 48-byte response is returned from iButton
- Created application to set each single-byte password and monitor serial port for response
- Trial and error to determine how response was generated for longer length passwords

# iButton DS1991 4

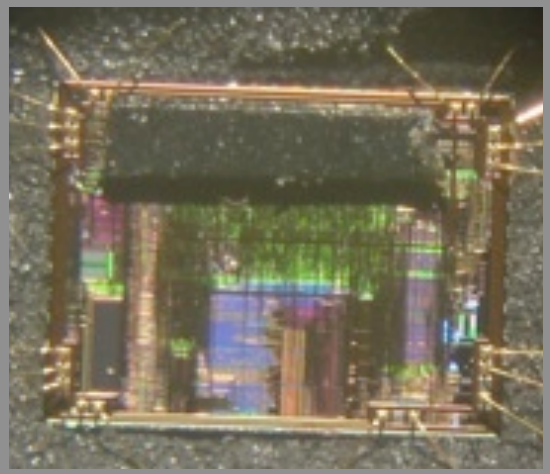
- Based on input password and 12kB constant block
  - Constant for all DS1991 devices
- Can precompute the 48-byte return value expected for an incorrect password
  - "DS1991 MultiKey iButton Dictionary Attack Vulnerability," `www.grandideastudio.com/portfolio/ds1991-ibutton-dictionary-attack/`
- If return value does not match, must be the correct password and subkey data

# Microchip PIC Config. Fuses

- Configuration fuses (including code protection bit) can be erased from some devices with UV light
  - "Hacking the PIC18F1320," `www.bunniestudios.com/wordpress/?page_id=40`
- Flash floating-gate transistor structures similar to UV-erasable



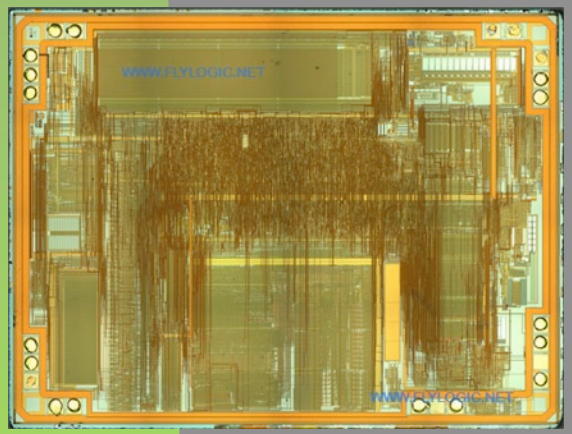

→



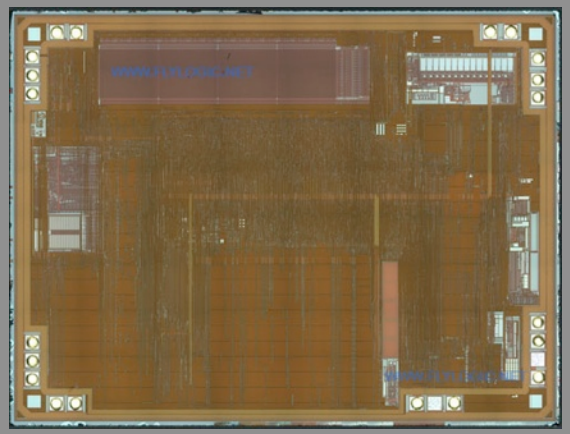
→

# Microchip PIC Config. Fuses 2

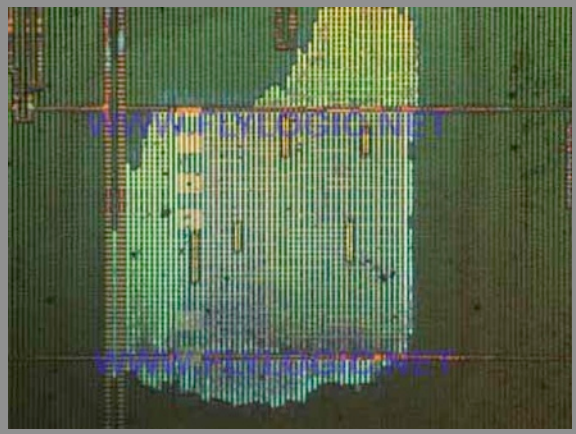
- Microchip revised die with additional metal fill
- Makes the attack slightly more difficult, but not impossible:
  - "Unmarked die revisions: Part I," `www.flylogic.net/blog/?p=9`
  - "Unmarked die revisions: Part II," `www.flylogic.net/blog/?p=12`




→



→



# MSP430 Serial Bootstrap Loader

- "MSP430 Security," `http://frob.us/projects/msp430hack/`
- Used for firmware reading/writing
  - BSL located in masked ROM portion of MSP430
  - Although MSP430 JTAG can be disabled by physically blowing fuse, BSL cannot
  - When enabled, BSL is protected by 32-byte password

# MSP430 Serial Bootstrap Loader 2

- The password comparison routine has unbalanced timing (v2.12)
  - An incorrect byte takes 2 clock cycles longer than a correct byte
  - So, present a password and monitor the timing
  - Can greatly reduce the amount of time required for a brute-force attack by determining how many bytes of the password are correct

# Boston MBTA CharlieCard/Ticket

- Stored value and/or time-based pass (unlimited rides during a given time period)
- CharlieTicket: Magnetic stripe, can be rewritten for value up to $655.36 by changing 16-bits corresponding to value
- CharlieCard: RFID-based smartcard using MIFARE Classic
  - Weak encryption leading to key recovery and full access to card
  - MIFARE Classic proprietary Crypto-1 algorithm previously broken by Karsten Nohl, et. al. 2007-2008 (`www.cs.virginia.edu/~kn5f/`)

# Boston MBTA CharlieCard/Ticket 2

- MBTA launched assault on researchers to try and squelch release of information prior to DEFCON 16 presentation
  - Only temporarily successful
  - Highlights the rift between security researchers/ hackers and vendors
- Ex.: `http://tech.mit.edu/V128/N30/subway.html`
- Ex.: `www.eff.org/cases/mbta-v-anderson`

# 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 to analyze San Francisco implementation
- Full details at `www.grandideastudio.com/portfolio/smart-parking-meters/`

# Parking Meter Technology

- Pure mechanical replaced with hybrid electromechanical in early 1990s
  - Mechanical coin slot
  - Minimal electronics used for timekeeping and administrator access (audit, debug, programming?)
- Now, we're seeing pure electronic "smart" systems
  - Microprocessor, memory, user interface
  - US is late to the game, other countries have been doing this for years

# Parking Meter Technology 2

- User Interfaces
  - Coin
  - Smartcard
  - Credit card
- Administrator Interfaces
  - Coin
  - Smartcard
  - Infrared
  - Wireless (RF, GPRS)
  - Other (Serial via key, etc.)

# Prior Problems and/or Failures

- New York City reset via infrared (universal remote control), 2001, `http://tinyurl.com/mae3g8`
- San Diego stored value card by H1kari, 2004, `www.uninformed.org/?v=1&a=6&t=txt`
- Chicago multi-space failures, June 2009
  - Firmware bug or intentional social disobedience?
  - `http://tinyurl.com/nt7gl9`
- Lots of other smartcard hacking has been done in the past
  - Ex.: Dutch phone cards (Hack-Tic), FedEx/ Kinko's, satellite TV (DirecTV/DISH)

# San Francisco MTA

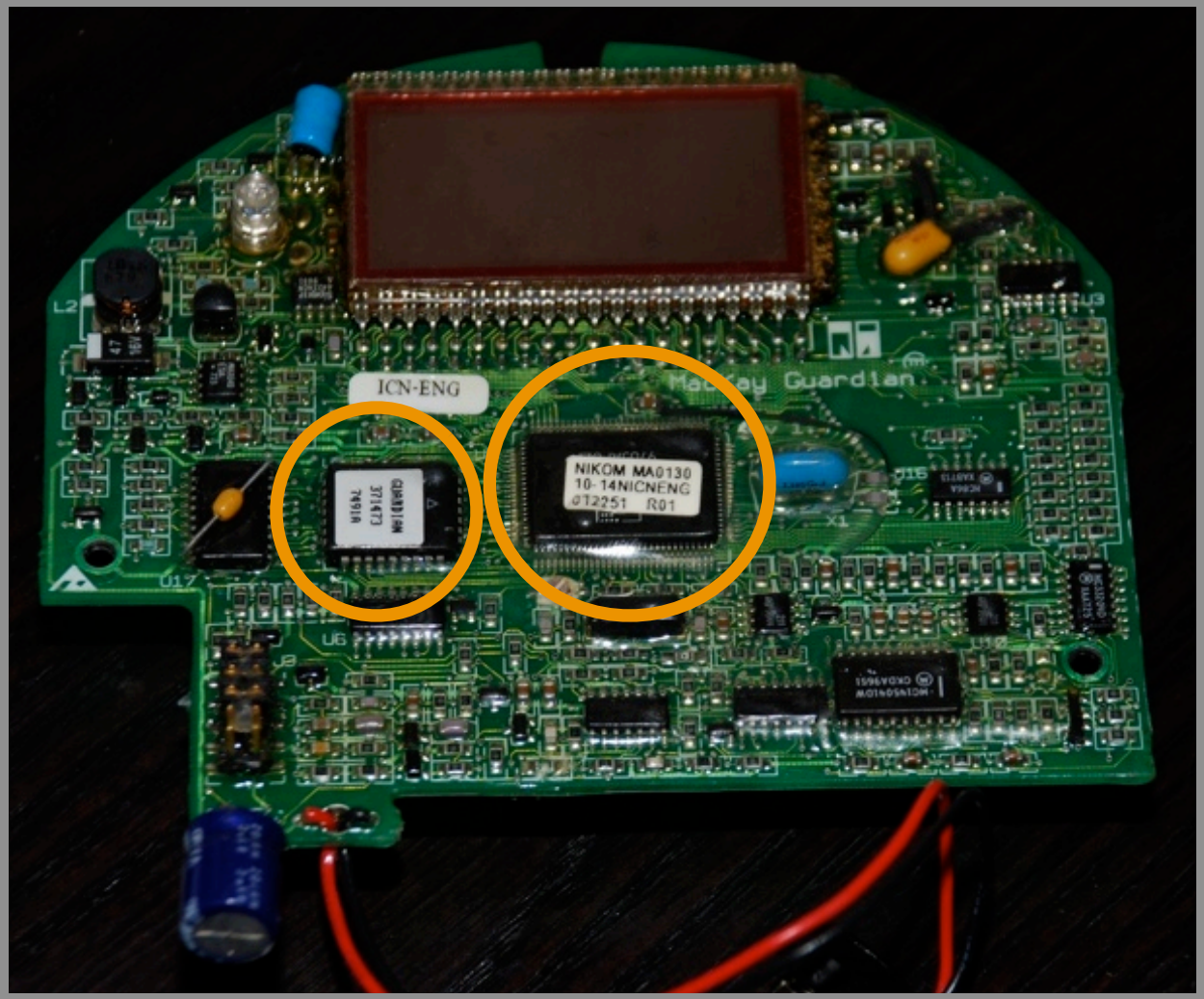
- Part of a $35 million pilot program to replace 23,000 mechanical meters with "smart" parking meters in 2003
- Infrastructure currently comprised of MacKay Guardian XLE meters
- Stored value smart card
  - $20 or $50 quantities
  - Can purchase online w/ credit card or in cash from selected locations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

# San Francisco MTA 2

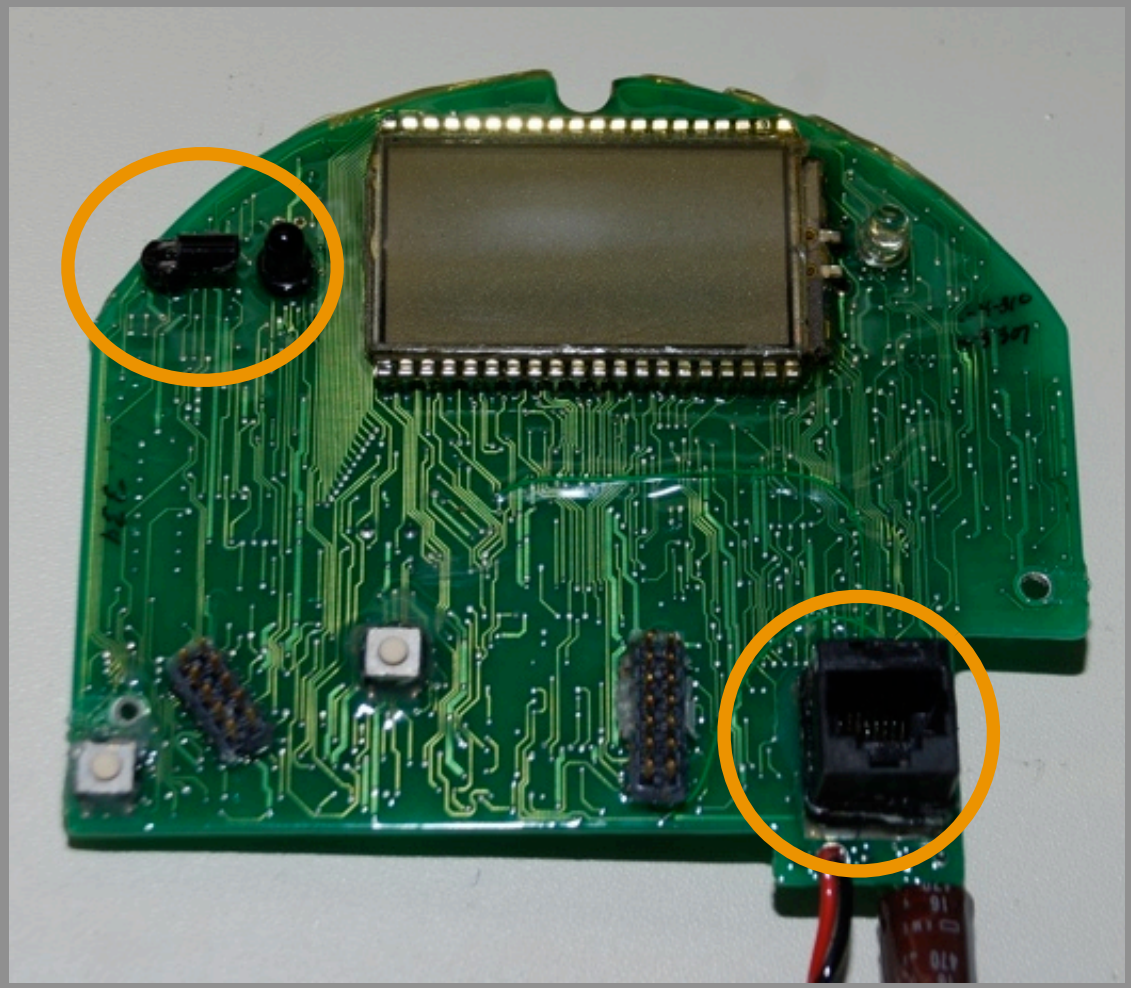
- Easy to replay transaction w/ modified data to obtain unlimited parking
  - Determined solely by looking at oscilloscope captures of smartcard communications
  - Succeeded in three days


# Meter Disassembly: MacKay Guardian


# Meter Disassembly: MacKay Guardian 2



# Meter Disassembly: MacKay Guardian 3



# 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
- How It's Made, Season 5, Episode 7: **`www.youtube.com/watch?v=1jzEcblRLEI`**
- Crawling the Internet for specific information
  - Product specifications, design documents, etc.

# 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

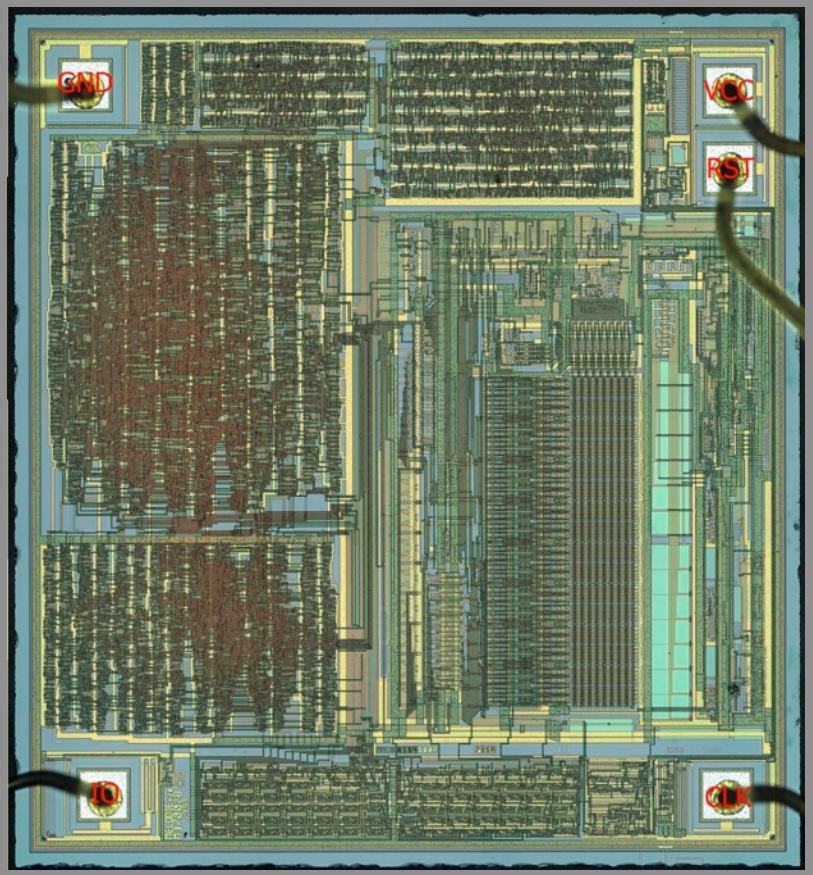
# Smartcard Die Analysis

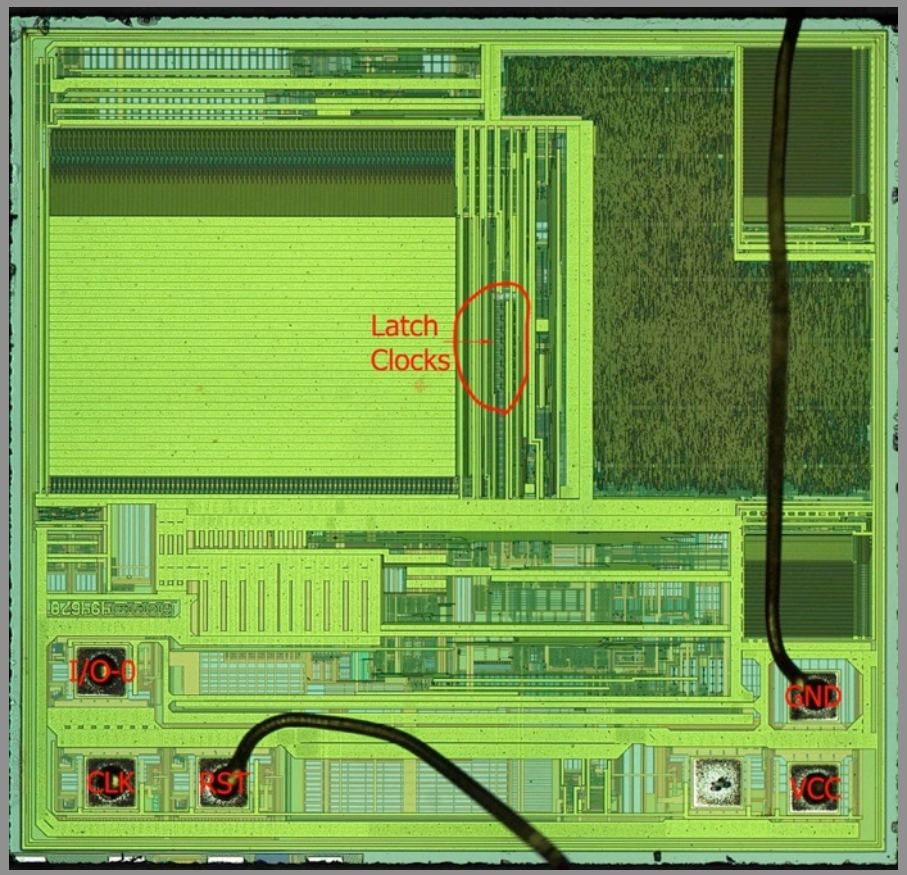
- Purchased and decapsulated multiple cards to look for clues of manufacturer and functionality
- Decapsulation process for smartcards
  1. Remove plastic surrounding the die (usually w/ acetone)
  2. Place die into small Pyrex of heated Fuming Nitric Acid ($HNO_3$)
  3. Rinse in acetone
  4. Glue die into a ceramic DIP package (for probing)
  5. If part is for analysis, prevent scratching!

# Smartcard Die Analysis 2

- 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)

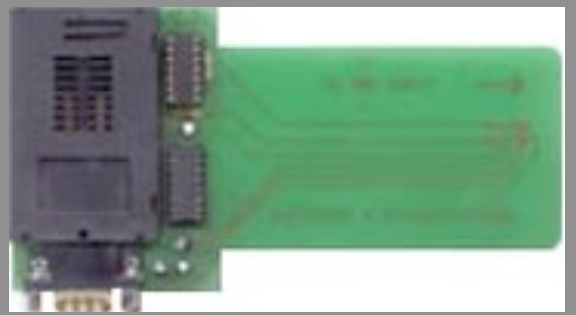
# Smartcard Die Analysis 3



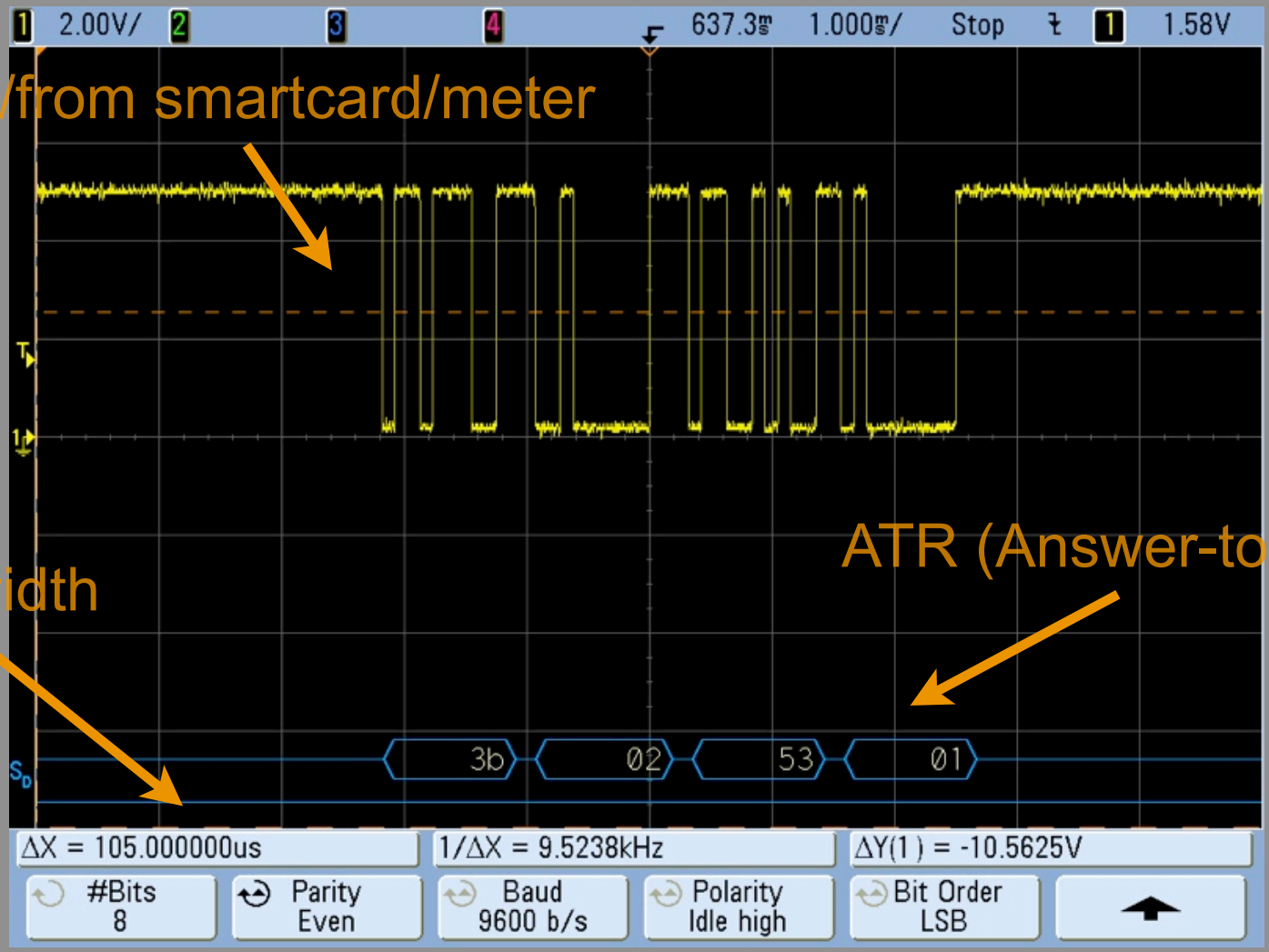



# Smartcard Communications Monitoring

- 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!

# Initialization

| *Meter* | | *Card* |
|---|---|---|
| Reset | → | |
| | ← | ATR |
| Read Address 0 | → | |
| | ← | Manufacturer ID |
| Read Address 1 | → | |
| | ← | Serial # |
| Read Address 2 | → | |
| | ← | Constant |
| Read Address 3 | → | |
| | ← | Unknown (8) |
| Read Address 4 | → | |
| | ← | [Used for meter to calculate CSC1 password?] |

[4 byte responses unless noted]

# Initialization 2

| *Meter* | | *Card* |
|---|---|---|
| | | [4 byte responses unless noted] |
| Read CSC1<br>Ratification Counter | → | 0 |
| CSC1 Password<br>[Password calculated by meter and sent to card for authentication] | → | Password OK (2) |
| Read Address 14 | → | 0 |
| Read CTC1<br>Card Transaction Counter | → | CTC1 [value varies] |

# Initialization 3

*Meter*

Read Balance 2

*Card*

[4 byte responses unless noted]

Maximum Card Value

Ex.: 0xFF FF F0 AF = $20

Ex.: 0xFF FF F1 27 = $50

Read CTC1

Card Transaction Counter

CTC1 [value varies]

# Deduction of Single Unit ($0.25)

| *Meter* | | *Card* |
|---|---|---|
| | | [4 byte responses unless noted] |
| Update Balance 1<br>Current Value A1 | → | OK (2) |
| Update Balance 1<br>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

# 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

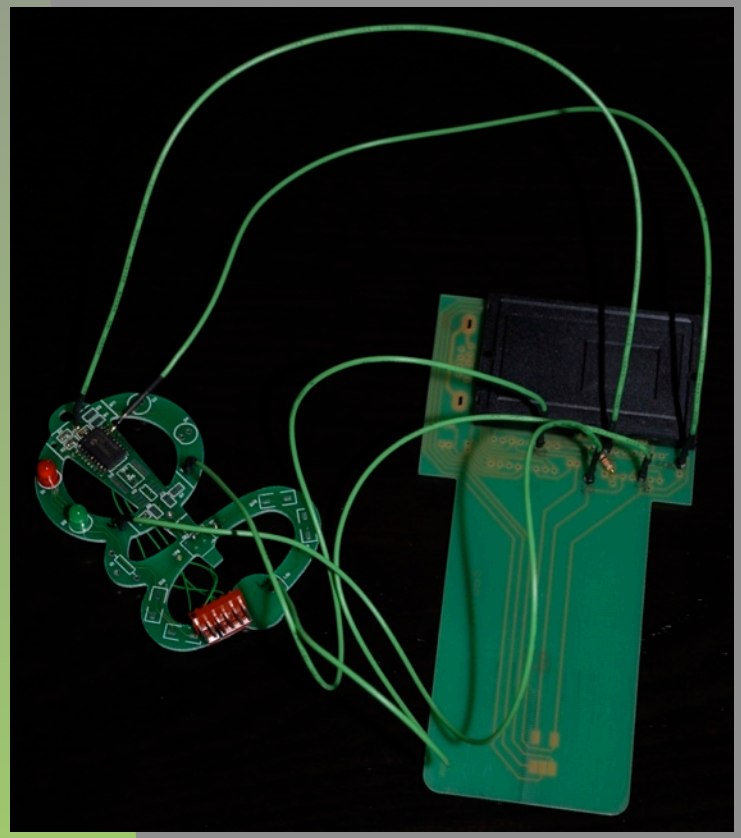
# Protocol Emulation 2

- Then, modified code to change various values until success
  - Knowing how "remaining value" is computed, what happens if we change Balance 2 to 0xFFF?
- Meter believes card has the maximum possible value

# Protocol Emulation 3

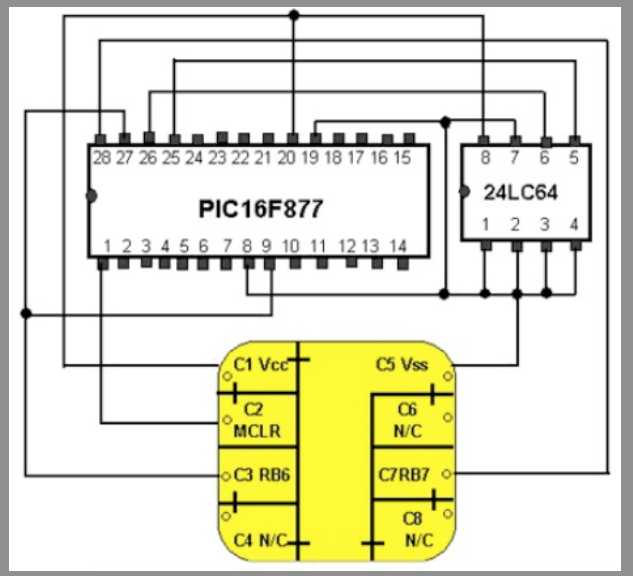
- Ported code to Silver Card (PIC16F877-based smartcard)
  - PIC-based smartcards have been popular for satellite TV hackers for years, so required tools are readily available
  - Ex.: `http://interesting-devices.com`

# 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
- Simplest attacks known for decades still work
- New skills and techniques continually being developed and shared
- Learn from history and other people's mistakes to make your products better!

# Q & A

[ joe @ grandideastudio.com ]