Tools of the Hardware Hacking Trade

Joe Grand (@joegrand)
Grand Idea Studio, Inc.

Finding the Right Tools for the Job

- Tools can help for design or "undesign"
- Access to tools is no longer a hurdle
- Can outsource to those with capabilities/equipment you don't have
- The key is knowing what tools are available and which one(s) are needed for a particular goal/attack

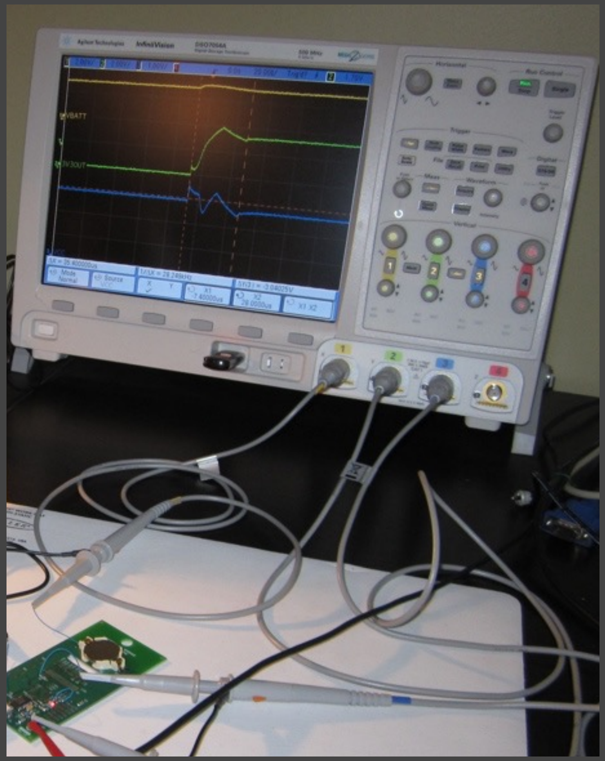
Tools of the Hardware Hacking Trade

- Signal Monitoring/Analysis
- Manipulation/Injection
- Imaging

Signal Monitoring / Analysis

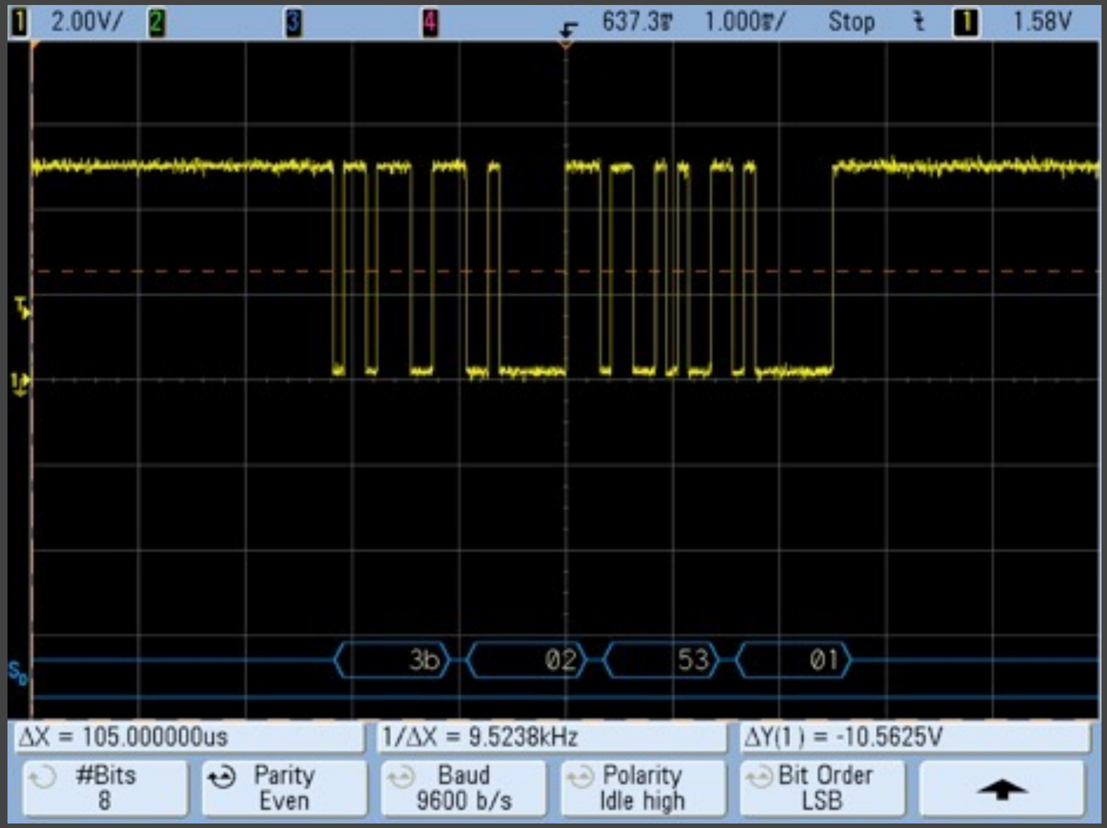
Oscilloscope

- Provides a visual display of electrical signals and how they change over time
- Introduction: **`www.tek.com/learning/oscilloscope-tutorial`**
- Range of hobbyist and professional tools
  - Analog/digital/mixed signal, # of channels (~1-8), bandwidth, sampling rate, resolution, buffer memory, trigger capabilities, math functions, protocol decoding, probe types, accessories
- Standalone: HP/Agilent, Tektronix, Rohde & Schwarz, LeCroy, Rigol
- PC-based: USBee, PicoScope, BitScope, PropScope
- Open: sciPrime, Smartscope, **`www.opencircuits.com/Oscilloscope#Open_Source_Oscilloscopes`**



Oscilloscope: Example

- SFMTA Smart Parking Meter (2009)
  - Joe Grand, Chris Tarnovsky, Jake Appelbaum
  - Monitored meter/card communication w/ oscilloscope
    - Slight variation in signal voltage determined direction of data
  - Created custom Microchip PIC-based smartcard emulator
  - `www.grandideastudio.com/portfolio/smart-parking-meters`



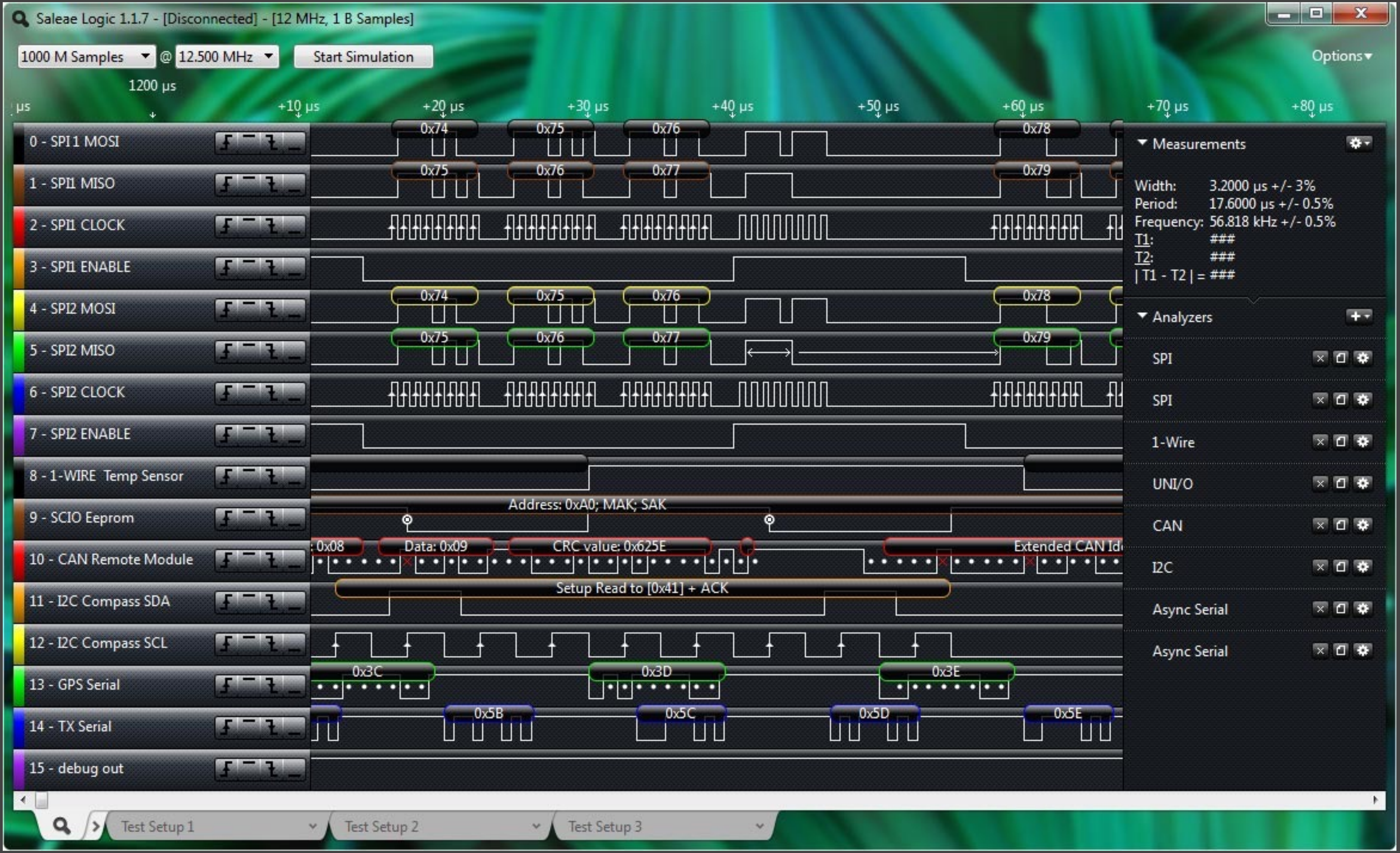
ΔX = 105.000000us
1/ΔX = 9.5238kHz
ΔY(1) = -10.5625V
#Bits 8
Parity Even
Baud 9600 b/s
Polarity Idle high
Bit Order LSB

Oscilloscope: Example 2

Logic Analyzer

- Used for concurrently capturing, visualizing, and decoding large quantities of digital data
- Introduction: `www.tek.com/learning/logic-analyzer-tutorial`
- Range of hobbyist and professional tools
  - # of channels (~>4), sampling rate, buffer memory, trigger capabilities, protocol decoding, probe types, accessories
- Standalone: HP/Agilent, Tektronix
- PC-based: Saleae Logic, LogicPort, USBee, LeCroy LogicStudio, DigiView
- Open: sigrok, Open Bench Logic Sniffer

Logic Analyzer 2



Saleae Logic 1.1.7 - [Disconnected] - [12 MHz, 1 B Samples]

1000 M Samples @ 12.500 MHz Start Simulation

Measurements

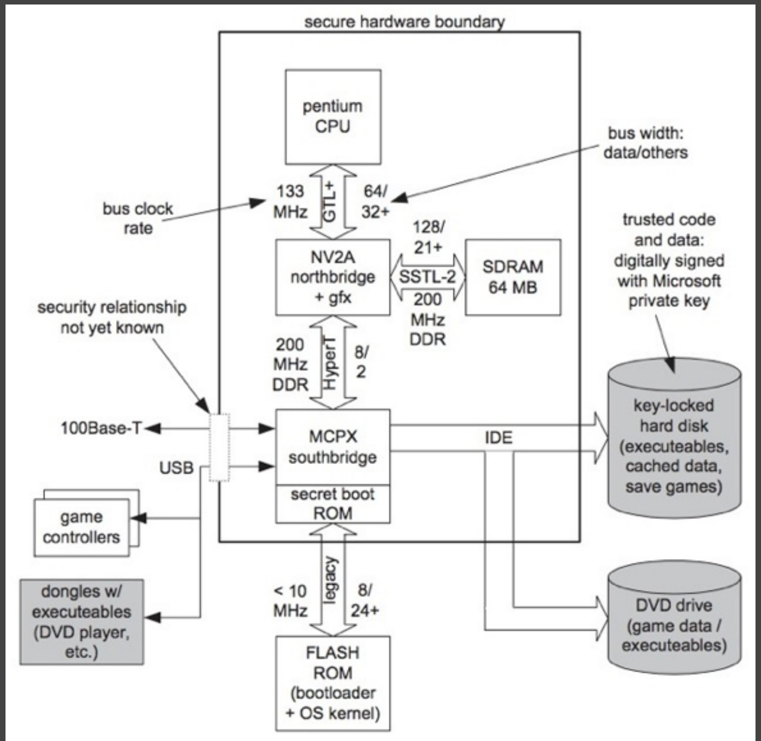
Width: 3.2000 µs +/- 3%
Period: 17.6000 µs +/- 0.5%
Frequency: 56.818 kHz +/- 0.5%
T1: ###
T2: ###
| T1 - T2 | = ###

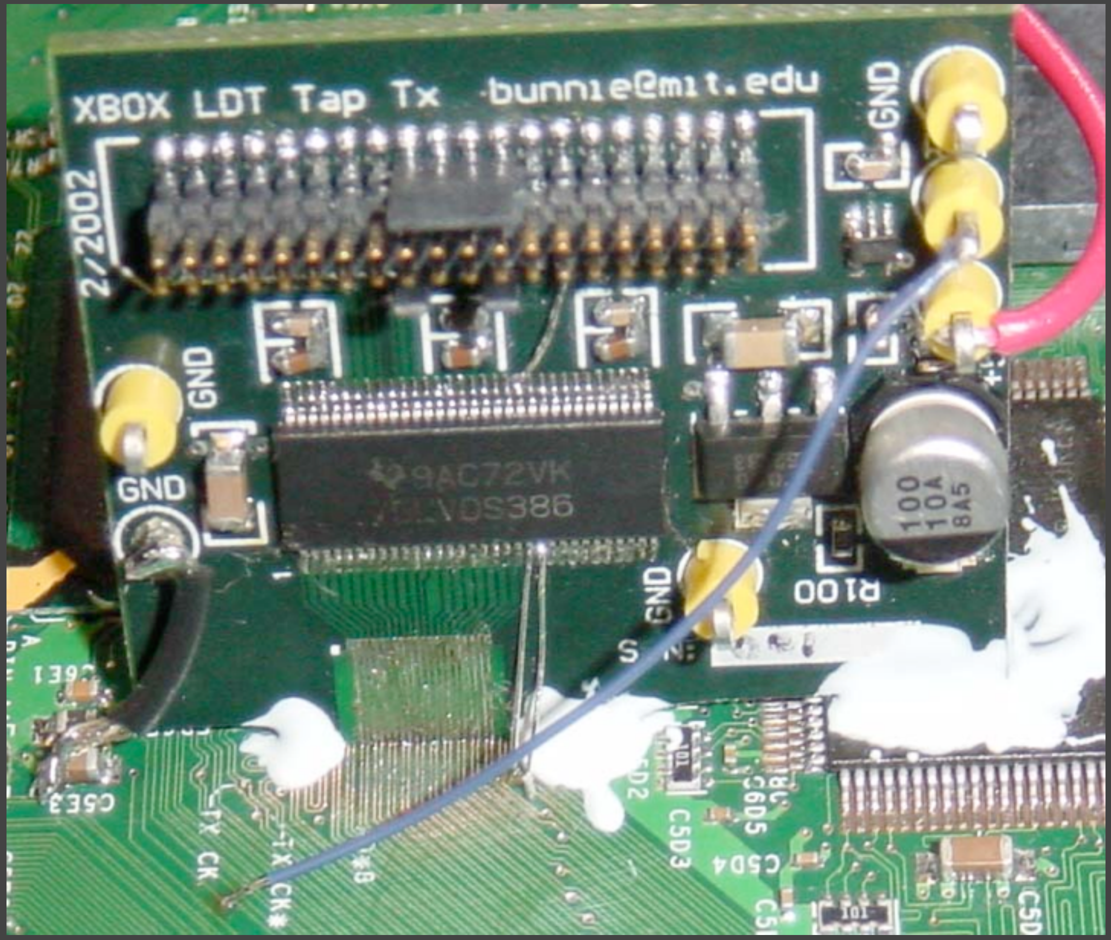
Analyzers

SPI
SPI
1-Wire
UNI/O
CAN
I2C
Async Serial
Async Serial

Logic Analyzer: Example

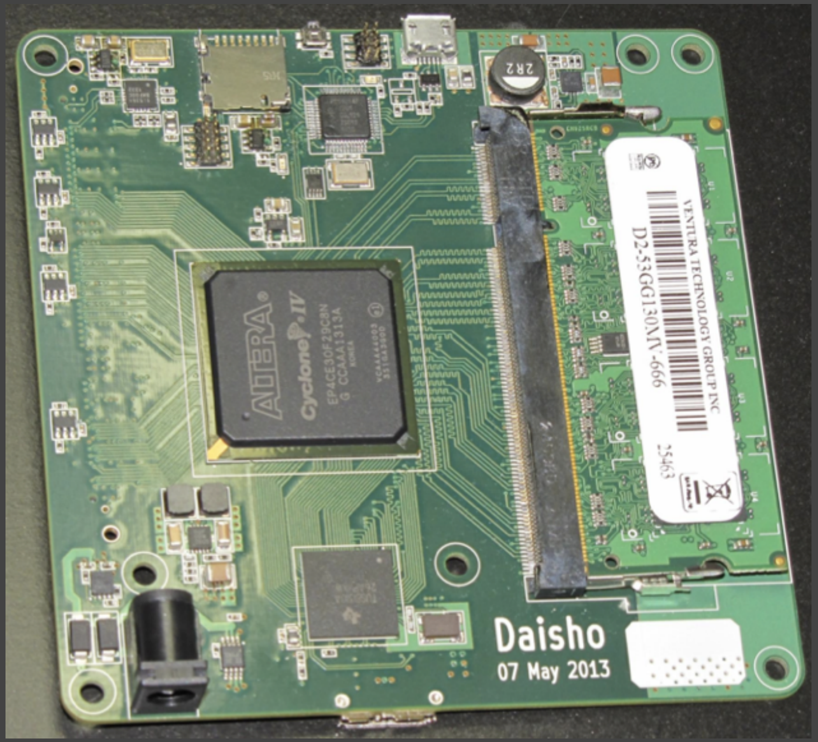
- Xbox (2002)
  - Bunnie Huang
  - Custom tap circuit to intercept secret boot loader over HyperTransport bus
  - Retrieved symmetric key from intercepted data to allow arbitrary code signing
  - `www.nostarch.com/xboxfree`


secure hardware boundary
pentium CPU
bus width: data/others
bus clock rate
133 MHz
GTL+
64/ 32+
128/ 21+
NV2A northbridge + gfx
SSTL-2
SDRAM 64 MB
200 MHz DDR
trusted code and data: digitally signed with Microsoft private key
security relationship not yet known
200 MHz DDR
HyperT
8/ 2
key-locked hard disk (executeables, cached data, save games)
100Base-T
USB
MCPX southbridge
IDE
secret boot ROM
game controllers
dongles w/ executeables (DVD player, etc.)
< 10 MHz
legacy
8/ 24+
DVD drive (game data / executeables)
FLASH ROM (bootloader + OS kernel)

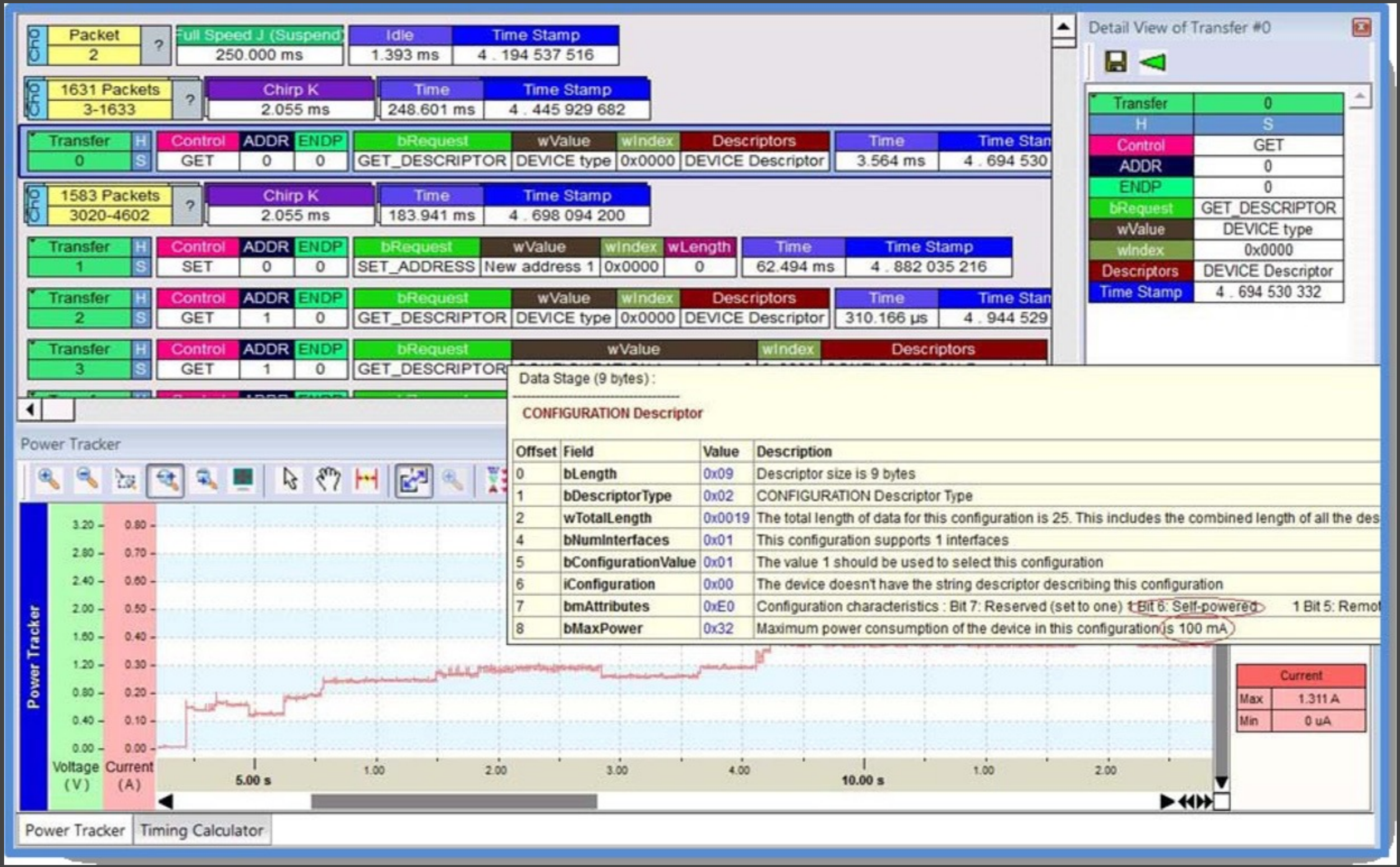


Protocol Analyzer

- Real-time, non-intrusive monitoring/capturing/decoding of wired communications
  - HW "man in the middle" to avoid any OS/SW overhead on host
  - Some also support data injection, power measurements
- Finisar Bus Doctor (Modular)
- Total Phase Beagle (USB/I2C/SPI) and Komodo (CAN)
- LeCroy Voyager (USB 2.0/3.0)
- Open: OpenVizsla, Daisho



Protocol Analyzer: Example



Packet	Full Speed J (Suspend)	Idle	Time Stamp
2	250.000 ms	1.393 ms	4 . 194 537 516

1631 Packets	Chirp K	Time	Time Stamp
3-1633	2.055 ms	248.601 ms	4 . 445 929 682

Transfer	H S	Control	ADDR	ENDP	bRequest	wValue	wIndex	Descriptors	Time	Time Stan
0		GET	0	0	GET_DESCRIPTOR	DEVICE type	0x0000	DEVICE Descriptor	3.564 ms	4 . 694 530

1583 Packets	Chirp K	Time	Time Stamp
3020-4602	2.055 ms	183.941 ms	4 . 698 094 200

Transfer	H S	Control	ADDR	ENDP	bRequest	wValue	wIndex	wLength	Time	Time Stamp
1		SET	0	0	SET_ADDRESS	New address 1	0x0000	0	62.494 ms	4 . 882 035 216

Transfer	H S	Control	ADDR	ENDP	bRequest	wValue	wIndex	Descriptors	Time	Time Stan
2		GET	1	0	GET_DESCRIPTOR	DEVICE type	0x0000	DEVICE Descriptor	310.166 µs	4 . 944 529

Transfer	H S	Control	ADDR	ENDP	bRequest	wValue	wIndex	Descriptors
3		GET	1	0	GET_DESCRIPTOR			

Detail View of Transfer #0

Transfer	0
H	S
Control	GET
ADDR	0
ENDP	0
bRequest	GET_DESCRIPTOR
wValue	DEVICE type
wIndex	0x0000
Descriptors	DEVICE Descriptor
Time Stamp	4 . 694 530 332

Data Stage (9 bytes):

CONFIGURATION Descriptor

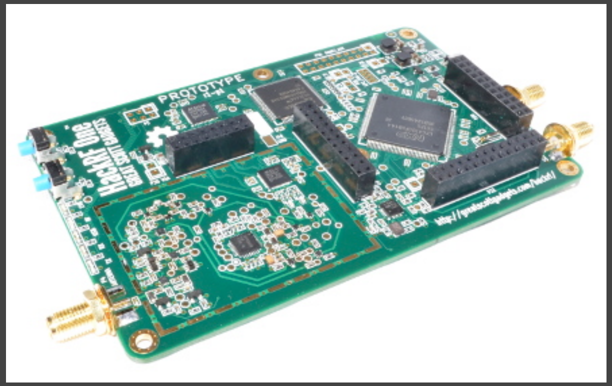
Offset	Field	Value	Description
0	bLength	0x09	Descriptor size is 9 bytes
1	bDescriptorType	0x02	CONFIGURATION Descriptor Type
2	wTotalLength	0x0019	The total length of data for this configuration is 25. This includes the combined length of all the des
4	bNumInterfaces	0x01	This configuration supports 1 interfaces
5	bConfigurationValue	0x01	The value 1 should be used to select this configuration
6	iConfiguration	0x00	The device doesn't have the string descriptor describing this configuration
7	bmAttributes	0xE0	Configuration characteristics : Bit 7: Reserved (set to one) 1 Bit 6: Self-powered 1 Bit 5: Remot
8	bMaxPower	0x32	Maximum power consumption of the device in this configuration is 100 mA

Current	
Max	1.311 A
Min	0 uA

http://teledynelecroy.com/doc/usb-voyager-m3i-power-tracker-usage-case-examples

Software Defined Radio

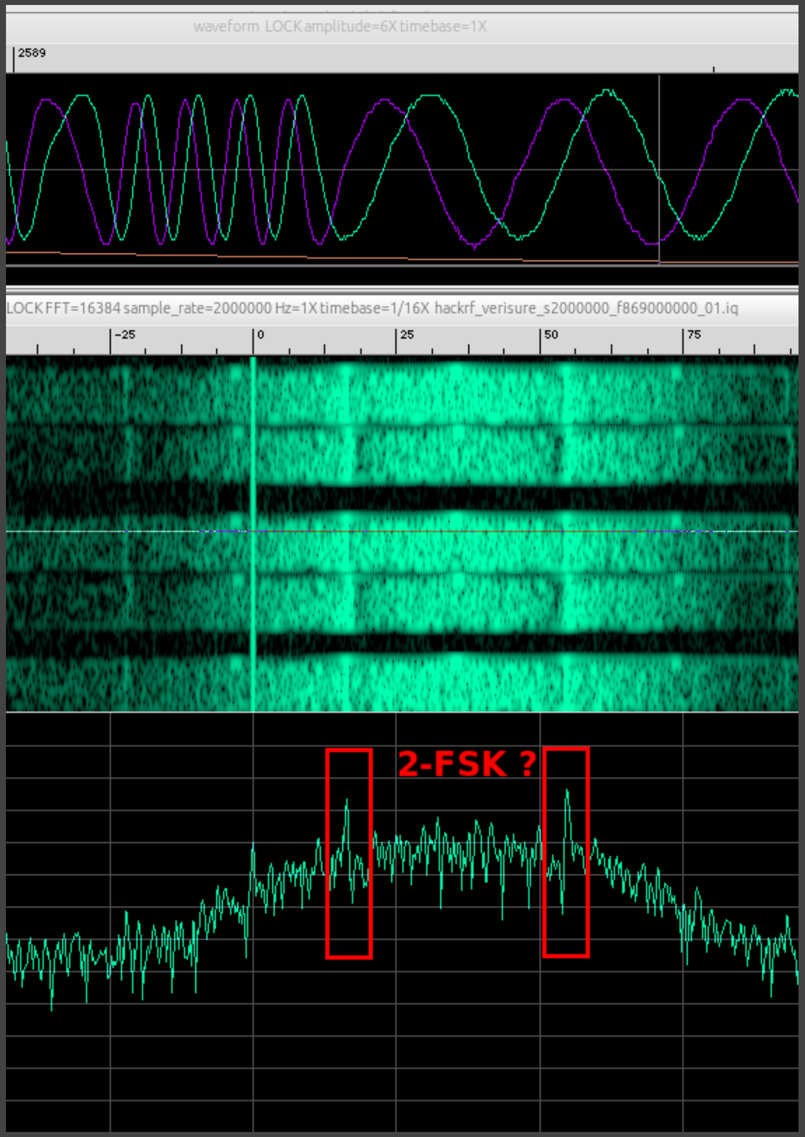
- Communication system where digital signal processing is used to implement radio/RF functions
  - Ex.: Mixers, filters, amplifiers, modulators/demodulators, detectors
  - RF front end + general purpose computer to receive/transmit arbitrary radio signals
- Primary toolset for RF/radio hacking
  - Visualize RF spectrum (spectrum analyzer)
  - Modulate/demodulate/filter raw signal
  - Decode/inject data
- Ex.: RTL-SDR, HackRF One, Blade RF, Ettus Research

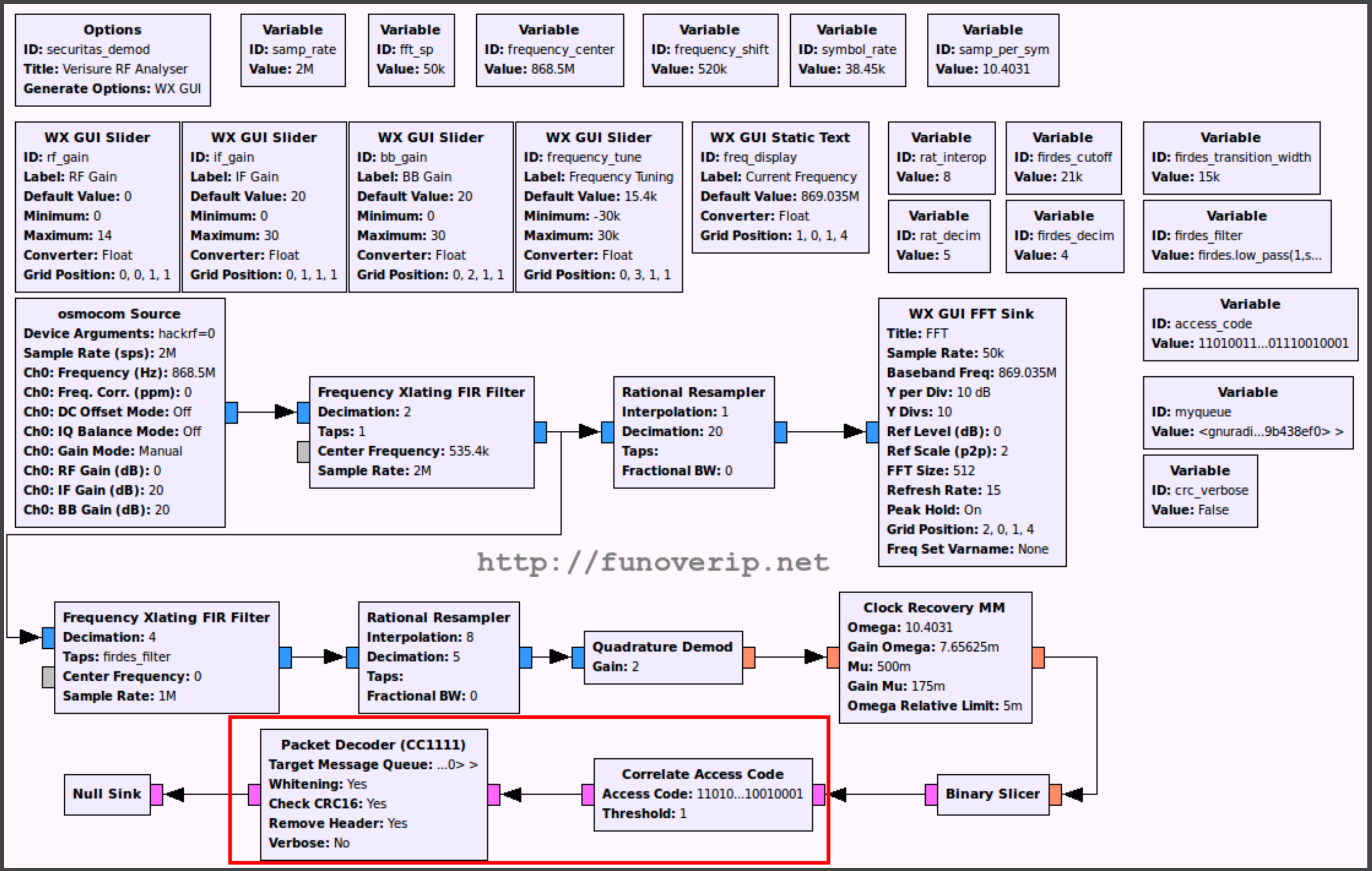


Software Defined Radio: Example

- Verisure Wireless Home Alarm
  - Discover frequency and modulation scheme using GQRX and HackRF
  - Capture raw signal and import into Baudline for visualization
  - Create custom flowgraph using GNU Radio to capture, filter, demodulate, and slice signal into binary data
  - `https://funoverip.net/2014/11/reverse-engineer-a-verisure-wireless-alarm-part-1-radio-communications/`

Software Defined Radio: Example 2





Options
ID: securitas_demod
Title: Verisure RF Analyser
Generate Options: WX GUI

Variable
ID: samp_rate
Value: 2M

Variable
ID: fft_sp
Value: 50k

Variable
ID: frequency_center
Value: 868.5M

Variable
ID: frequency_shift
Value: 520k

Variable
ID: symbol_rate
Value: 38.45k

Variable
ID: samp_per_sym
Value: 10.4031

WX GUI Slider
ID: rf_gain
Label: RF Gain
Default Value: 0
Minimum: 0
Maximum: 14
Converter: Float
Grid Position: 0, 0, 1, 1

WX GUI Slider
ID: if_gain
Label: IF Gain
Default Value: 20
Minimum: 0
Maximum: 30
Converter: Float
Grid Position: 0, 1, 1, 1

WX GUI Slider
ID: bb_gain
Label: BB Gain
Default Value: 20
Minimum: 0
Maximum: 30
Converter: Float
Grid Position: 0, 2, 1, 1

WX GUI Slider
ID: frequency_tune
Label: Frequency Tuning
Default Value: 15.4k
Minimum: -30k
Maximum: 30k
Converter: Float
Grid Position: 0, 3, 1, 1

WX GUI Static Text
ID: freq_display
Label: Current Frequency
Default Value: 869.035M
Converter: Float
Grid Position: 1, 0, 1, 4

Variable
ID: rat_interop
Value: 8

Variable
ID: firdes_cutoff
Value: 21k

Variable
ID: firdes_transition_width
Value: 15k

Variable
ID: rat_decim
Value: 5

Variable
ID: firdes_decim
Value: 4

Variable
ID: firdes_filter
Value: firdes.low_pass(1,s...

Variable
ID: access_code
Value: 11010011...01110010001

Variable
ID: myqueue
Value: <gnuradi...9b438ef0> >

Variable
ID: crc_verbose
Value: False

osmocom Source
Device Arguments: hackrf=0
Sample Rate (sps): 2M
Ch0: Frequency (Hz): 868.5M
Ch0: Freq. Corr. (ppm): 0
Ch0: DC Offset Mode: Off
Ch0: IQ Balance Mode: Off
Ch0: Gain Mode: Manual
Ch0: RF Gain (dB): 0
Ch0: IF Gain (dB): 20
Ch0: BB Gain (dB): 20

Frequency Xlating FIR Filter
Decimation: 2
Taps: 1
Center Frequency: 535.4k
Sample Rate: 2M

Rational Resampler
Interpolation: 1
Decimation: 20
Taps:
Fractional BW: 0

WX GUI FFT Sink
Title: FFT
Sample Rate: 50k
Baseband Freq: 869.035M
Y per Div: 10 dB
Y Divs: 10
Ref Level (dB): 0
Ref Scale (p2p): 2
FFT Size: 512
Refresh Rate: 15
Peak Hold: On
Grid Position: 2, 0, 1, 4
Freq Set Varname: None

http://funoverip.net

Frequency Xlating FIR Filter
Decimation: 4
Taps: firdes_filter
Center Frequency: 0
Sample Rate: 1M

Rational Resampler
Interpolation: 8
Decimation: 5
Taps:
Fractional BW: 0

Quadrature Demod
Gain: 2

Clock Recovery MM
Omega: 10.4031
Gain Omega: 7.65625m
Mu: 500m
Gain Mu: 175m
Omega Relative Limit: 5m

Binary Slicer

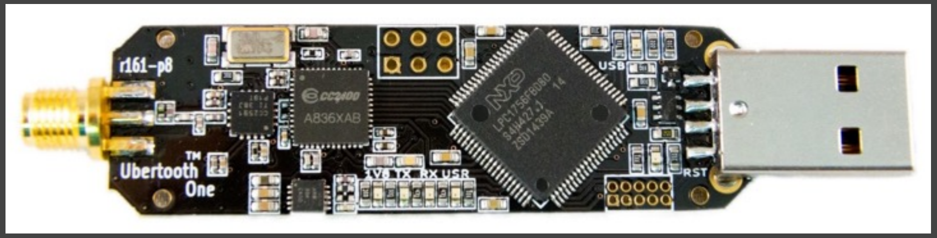
Correlate Access Code
Access Code: 11010...10010001
Threshold: 1

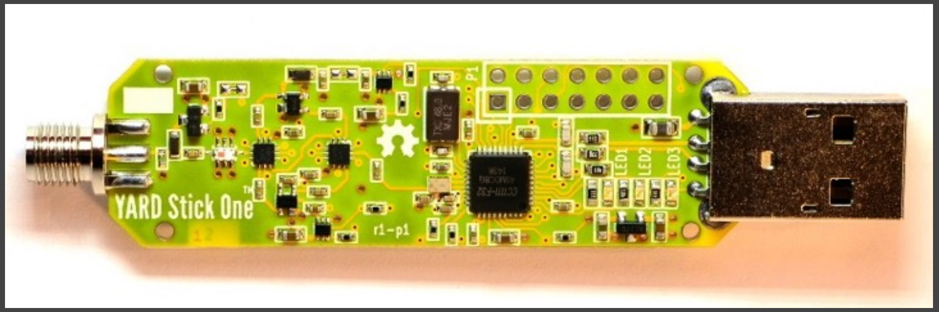
Packet Decoder (CC1111)
Target Message Queue: ...0> >
Whitening: Yes
Check CRC16: Yes
Remove Header: Yes
Verbose: No

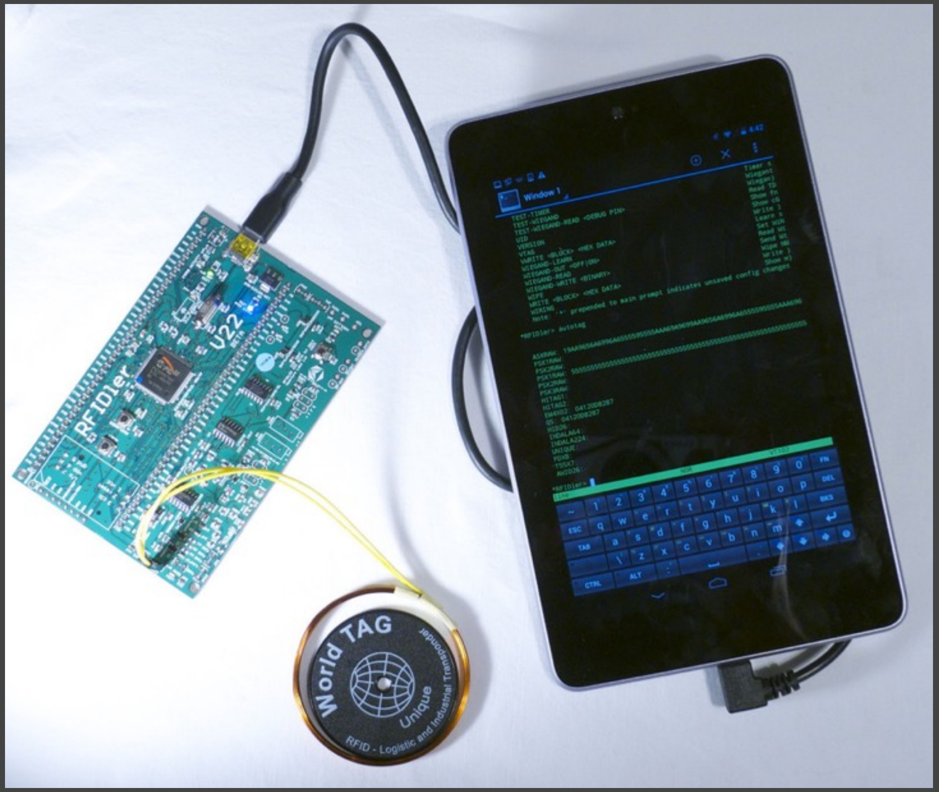
Null Sink

More Wireless

- Ubertooth One
  - Bluetooth/2.4GHz
- YARD Stick One
  - General purpose RF, < 1GHz
- WiFi Pineapple
  - Penetration testing/attacks
- Femtocell
  - Cellular data interception
- RaspBee
  - ZigBee module for Raspberry Pi
  - Command injection via custom firmware
- RFIDler
  - RFID reading/writing/emulation (125/134kHz)

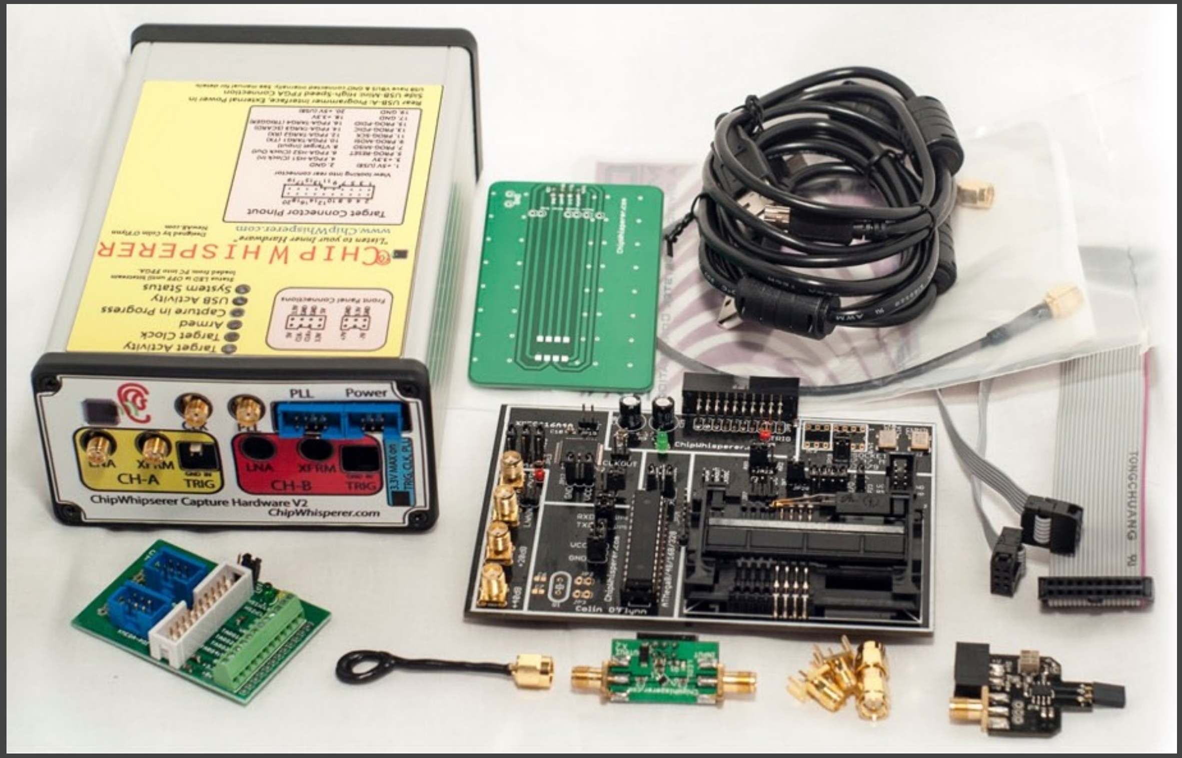


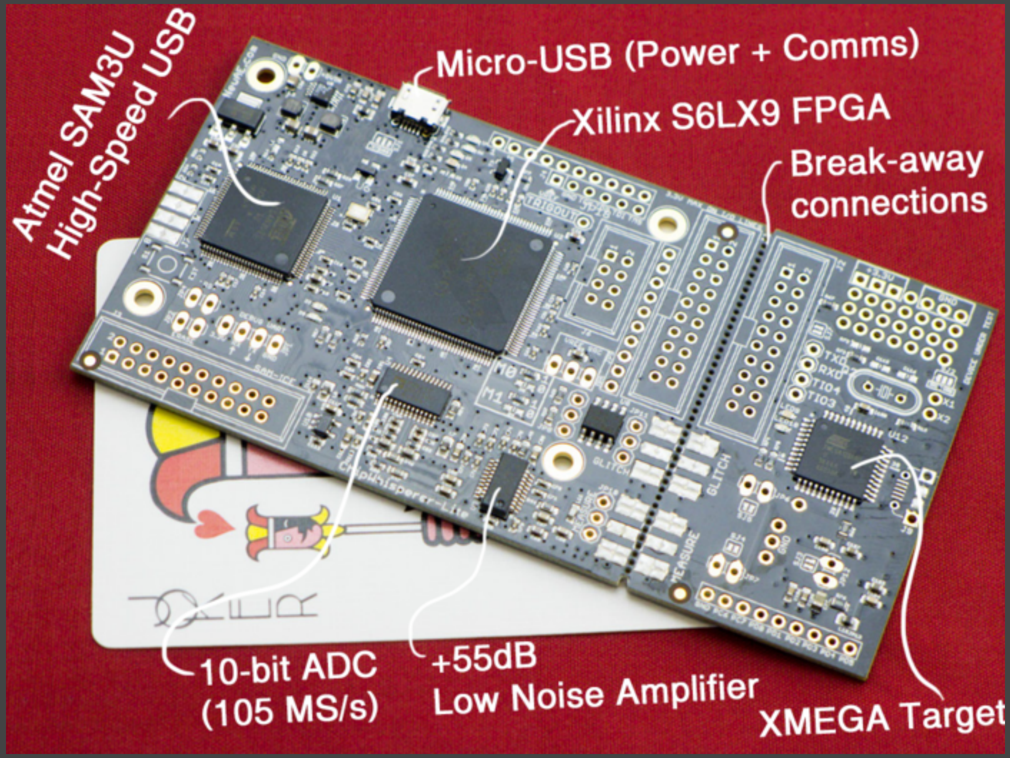




ChipWhisperer (and -Lite)

- Colin O'Flynn
- Collection of open source HW/SW tools for side channel, timing, and glitching attacks
- Supports AES-128/256 key extraction via EM/power analysis
  - Correlate measured power w/ predicted power to guess byte of key
- `www.chipwhisperer.com`

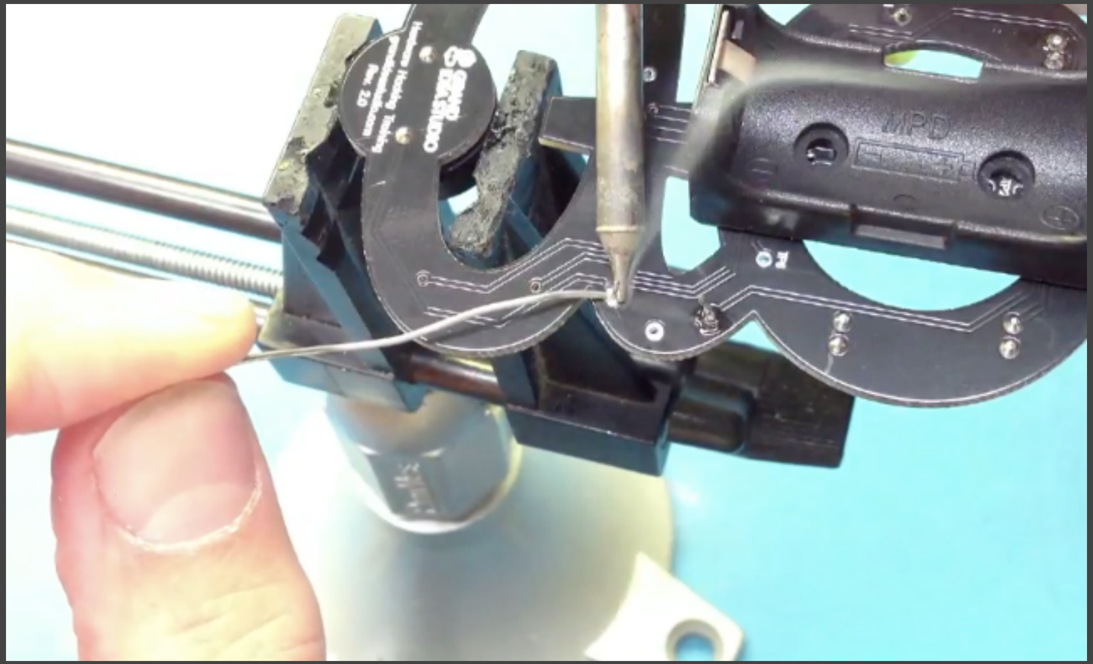



Atmel SAM3U High-Speed USB
Micro-USB (Power + Comms)
Xilinx S6LX9 FPGA
Break-away connections
10-bit ADC (105 MS/s)
+55dB Low Noise Amplifier
XMEGA Target

Manipulation / Injection

Soldering Iron

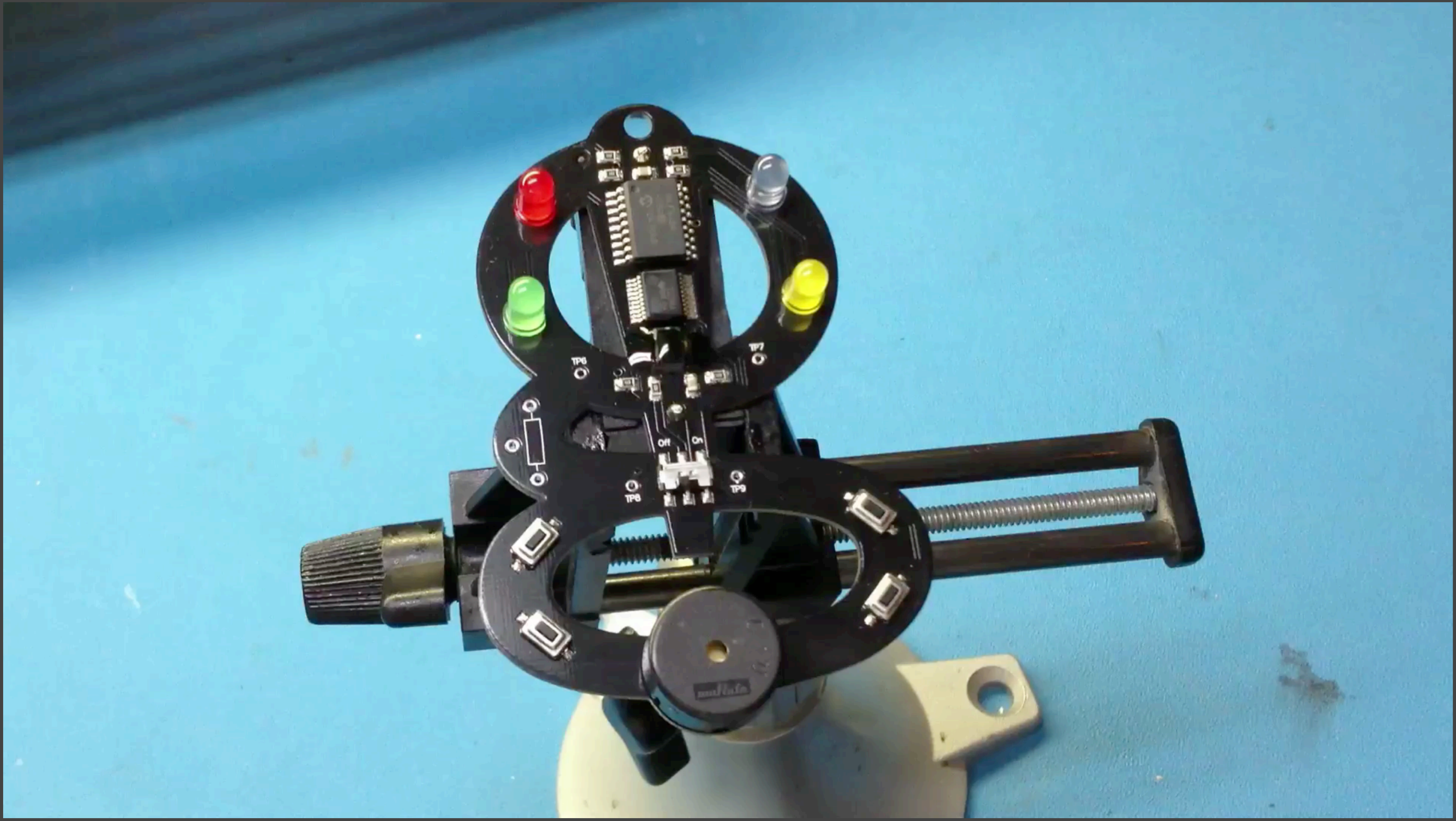
- Provides heat to melt solder that physically holds components on a circuit board
- Range from a simple stick iron to a full-fledged rework station
  - Interchangeable tips, adjustable temperature, hot air reflow
- Weller, Metcal, Hakko, Radio Shack (!)
- Open: Soldering Iron Driver Board, `http://dangerousprototypes.com/docs/Soldering_Iron_Driver`



ChipQuik

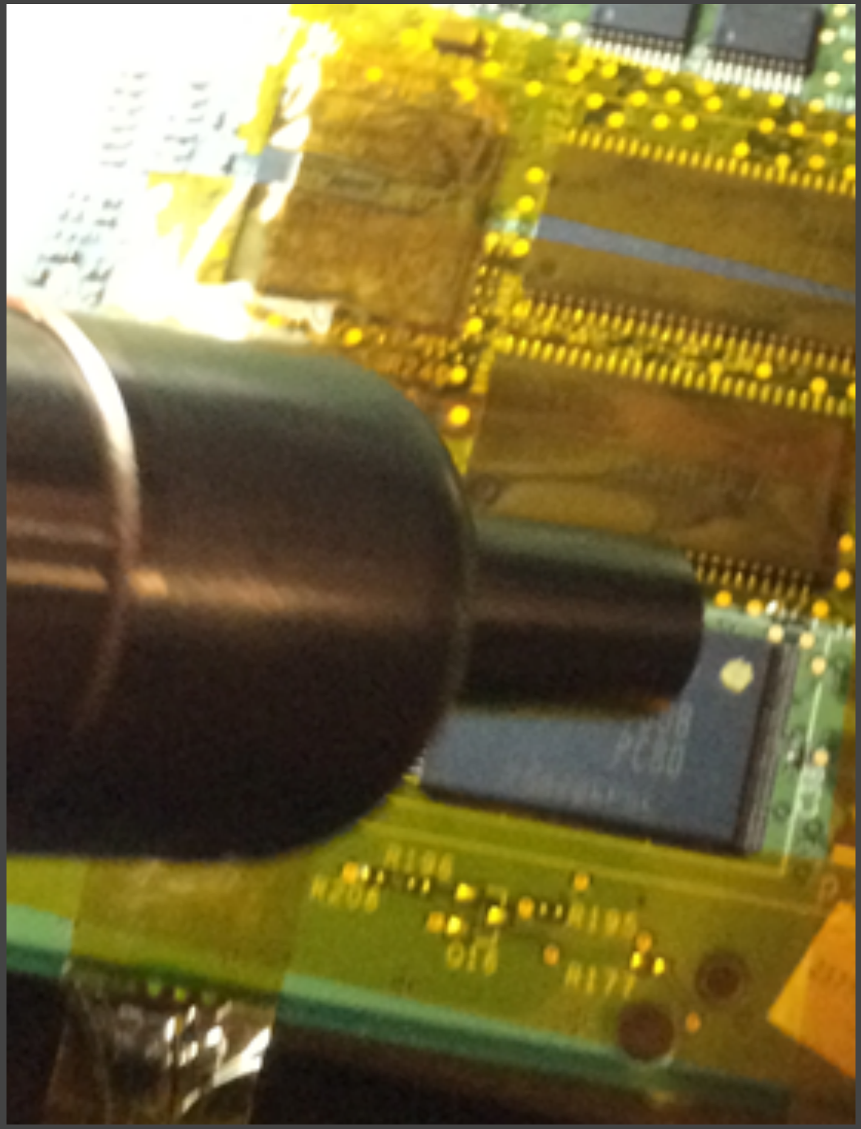
- Provides quick and easy removal of surface mount (and some through hole) components
- Primary component is a low-melting temperature alloy (< 200°F)
  - Reduces the overall melting temperature of the solder
  - Allows you to lift/slide the part of the board
- `www.chipquik.com`

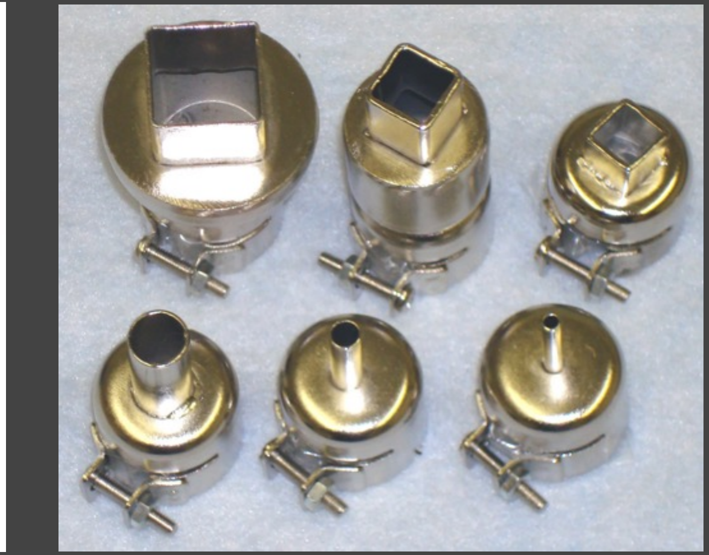
ChipQuik: Example



Rework Station

- Allows easier removal and reflow of individual SMD components (aka "chip off")
- Hot air convection
  - Most accessible, cost effective
  - Nozzles for different package types/ mechanical footprints
  - Difficult to focus heat on just the target component



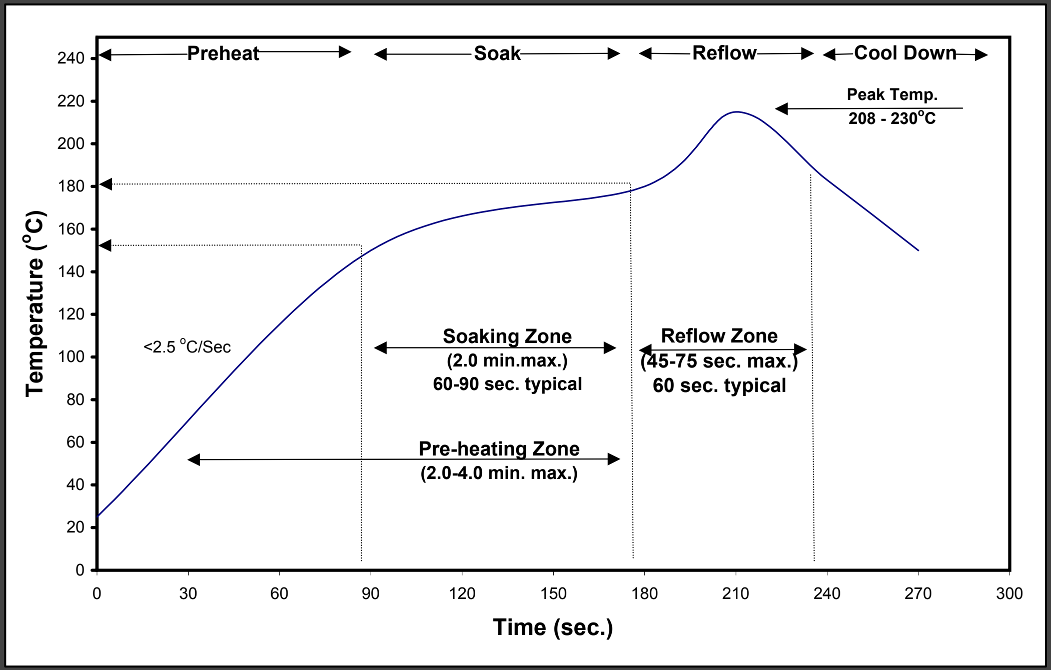


Rework Station 2

- Infrared and/or laser
  - More complex, expensive systems
  - Provides focused heat on specific component
  - Many are programmable for various heating profiles
- Beware of repeated thermal cycling, which could damage IC
- Ex.: Weller, Metcal, Hakko, ZEVAC, Zephyrtronics

Reflow Oven

- Follows recommended solder profile for PCB assembly (and disassembly)
- Closed loop PID for accurate temperature control
- Avoids damage to parts due to improper heating   and/or thermal cycling
- Ex.: T-962A, **`https://github.com/UnifiedEngineering/T-962-improvements`**

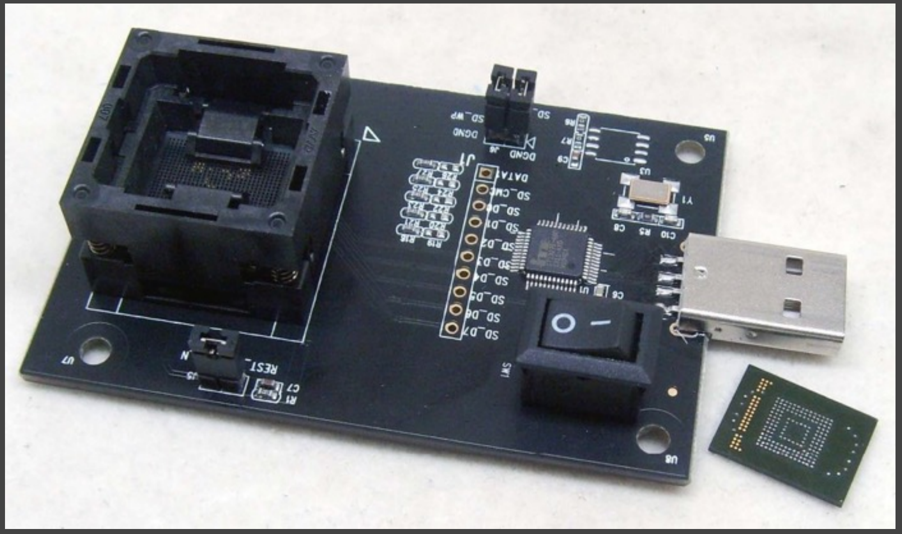


Reflow Oven 2

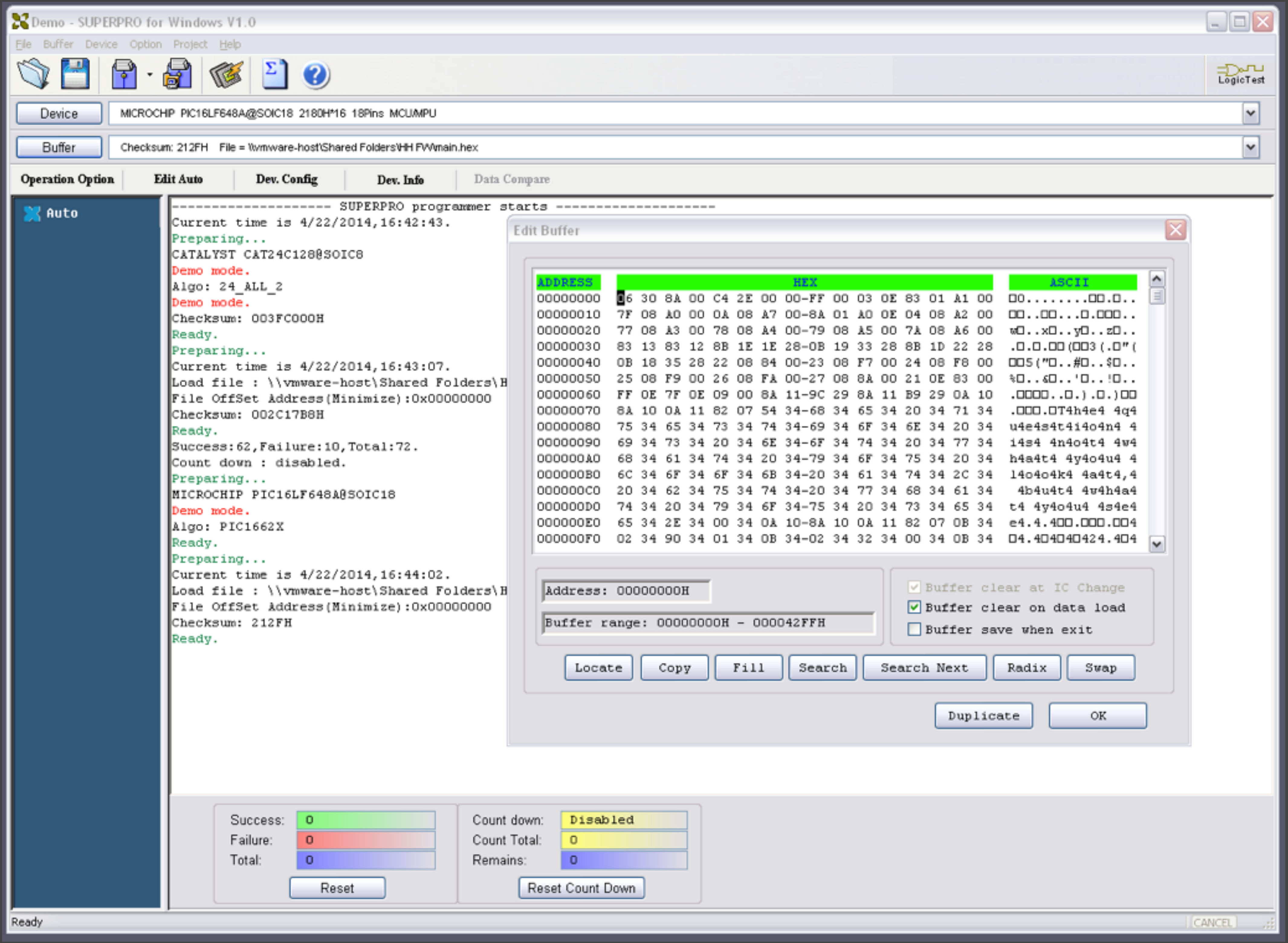
- Many toaster ovens can be hacked/modified using external controller
  - Ex.: Reflowster, Rocket Stream Controller Shield, Hobbybotics Reflow Controller, ControLeo2

Device Programmer

- Used to read/write most devices that contain memory
  - Standalone or internal to MCU
  - Ex.: Flash, E(E)PROM, ROM, RAM, PLD/CPLD, FPGA
- Many support > 90k (!) different devices
- Some devices can be manipulated in-circuit
- Few code protection mechanisms exist
  - Security bit/fuse, password
- EE Tools, Xeltek, BP Microsystems, Data IO, GALEP (open API)



Device Programmer: Example



Device: MICROCHIP PIC16LF648A@SOIC18 2180H*16 18Pins MCU/MPU

Buffer: Checksum: 212FH File = \\vmware-host\Shared Folders\HH FW\main.hex

```
-------------------- SUPERPRO programmer starts --------------------
Current time is 4/22/2014,16:42:43.
Preparing...
CATALYST CAT24C128@SOIC8
Demo mode.
Algo: 24_ALL_2
Demo mode.
Checksum: 003FC000H
Ready.
Preparing...
Current time is 4/22/2014,16:43:07.
Load file : \\vmware-host\Shared Folders\H
File OffSet Address(Minimize):0x00000000
Checksum: 002C17B8H
Ready.
Success:62,Failure:10,Total:72.
Count down : disabled.
Preparing...
MICROCHIP PIC16LF648A@SOIC18
Demo mode.
Algo: PIC1662X
Ready.
Preparing...
Current time is 4/22/2014,16:44:02.
Load file : \\vmware-host\Shared Folders\H
File OffSet Address(Minimize):0x00000000
Checksum: 212FH
Ready.
```

```
ADDRESS                         HEX                               ASCII
00000000  06 30 8A 00 C4 2E 00 00-FF 00 03 0E 83 01 A1 00  □0........□□.□..
00000010  7F 08 A0 00 0A 08 A7 00-8A 01 A0 0E 04 08 A2 00  □□..□□...□.□□□..
00000020  77 08 A3 00 78 08 A4 00-79 08 A5 00 7A 08 A6 00  w□..x□..y□..z□..
00000030  83 13 83 12 8B 1E 1E 28-0B 19 33 28 8B 1D 22 28  .□.□.□□(□□3(.□"(
00000040  0B 18 35 28 22 08 84 00-23 08 F7 00 24 08 F8 00  □□5("□..#□..$□..
00000050  25 08 F9 00 26 08 FA 00-27 08 8A 00 21 0E 83 00  %□..&□..'□..!□..
00000060  FF 0E 7F 0E 09 00 8A 11-9C 29 8A 11 B9 29 0A 10  .□□□□..□.).□.)□□
00000070  8A 10 0A 11 82 07 54 34-68 34 65 34 20 34 71 34  .□□□.□T4h4e4 4q4
00000080  75 34 65 34 73 34 74 34-69 34 6F 34 6E 34 20 34  u4e4s4t4i4o4n4 4
00000090  69 34 73 34 20 34 6E 34-6F 34 74 34 20 34 77 34  i4s4 4n4o4t4 4w4
000000A0  68 34 61 34 74 34 20 34-79 34 6F 34 75 34 20 34  h4a4t4 4y4o4u4 4
000000B0  6C 34 6F 34 6F 34 6B 34-20 34 61 34 74 34 2C 34  l4o4o4k4 4a4t4,4
000000C0  20 34 62 34 75 34 74 34-20 34 77 34 68 34 61 34   4b4u4t4 4w4h4a4
000000D0  74 34 20 34 79 34 6F 34-75 34 20 34 73 34 65 34  t4 4y4o4u4 4s4e4
000000E0  65 34 2E 34 00 34 0A 10-8A 10 0A 11 82 07 0B 34  e4.4.4□□.□□□.□□4
000000F0  02 34 90 34 01 34 0B 34-02 34 32 34 00 34 0B 34  □4.4□4□4□424.4□4
```

Address: 00000000H

Buffer range: 00000000H - 000042FFH

☑ Buffer clear at IC Change

☑ Buffer clear on data load

☐ Buffer save when exit

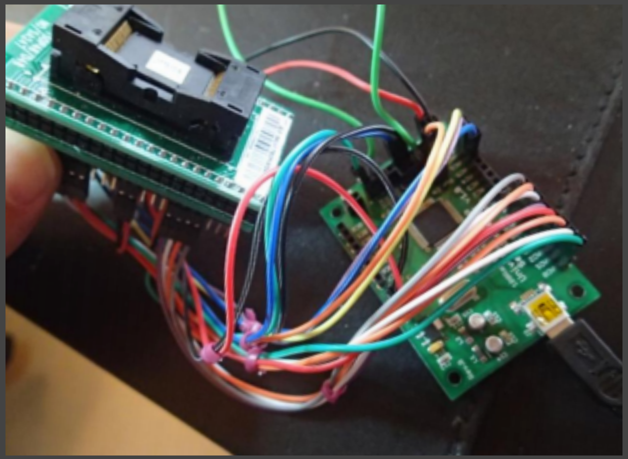
Success: 0
Failure: 0
Total: 0

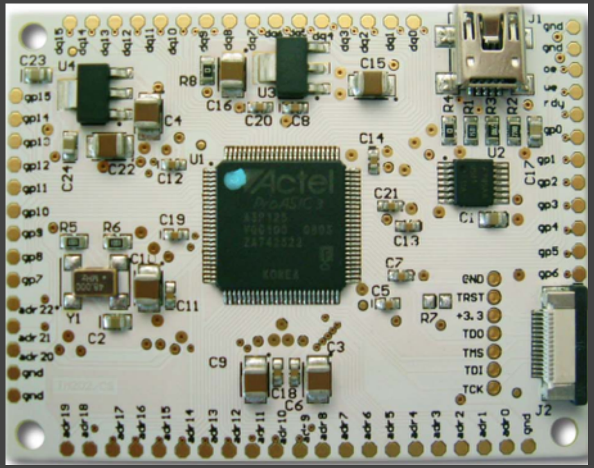
Count down: Disabled
Count Total: 0
Remains: 0

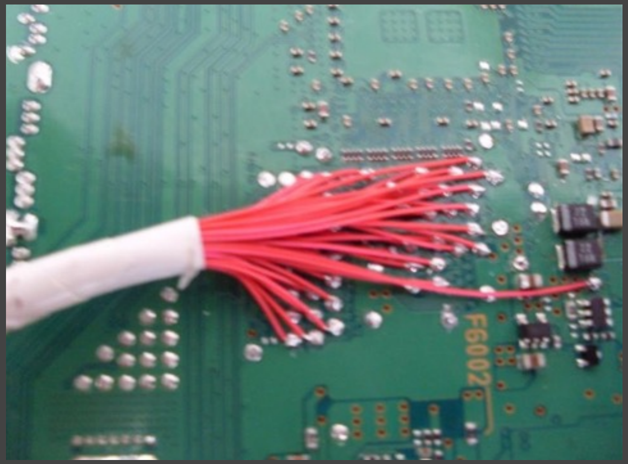
Ready

Device Programmer: Hacker Specific

- Arduino Parallel Flash Dumper, `https://github.com/cyphunk/ParallelFLASHDumper`
- flashrom, `http://flashrom.org`
- Infectus, Noraliser, NAND/NORway, Progskeet, PNM, PIC32MX, E3, `www.psdevwiki.com/ps3/Hardware_flashing`

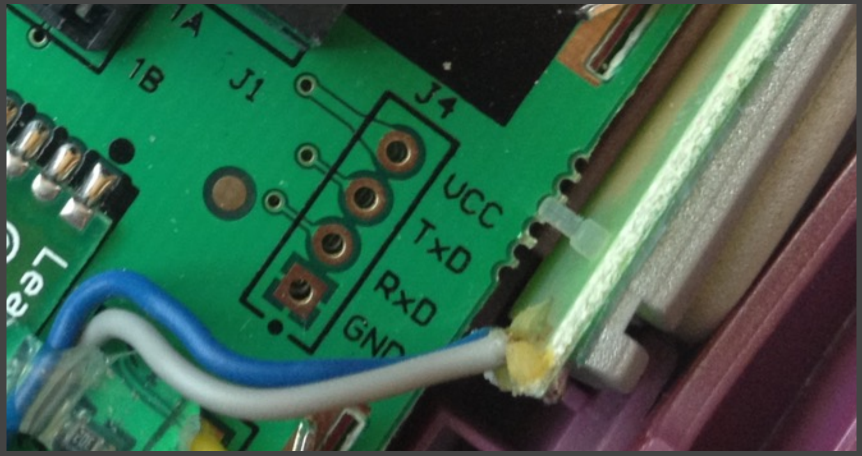


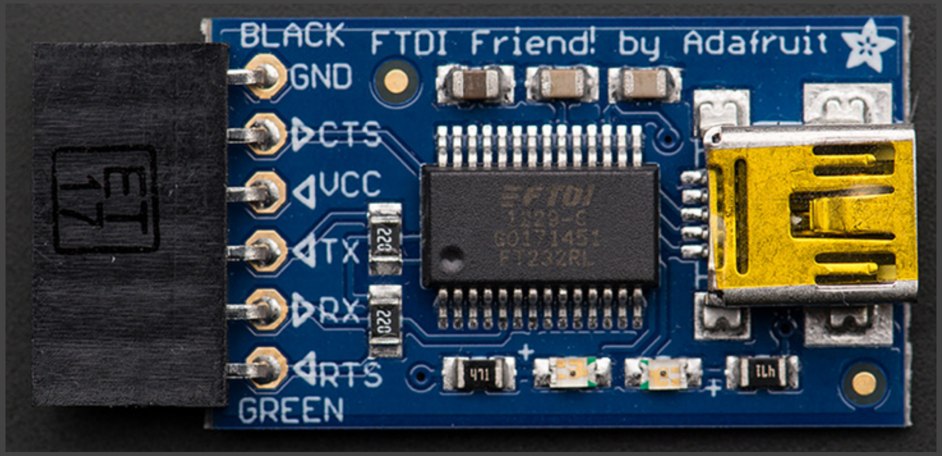




USB-to-Serial Adapter

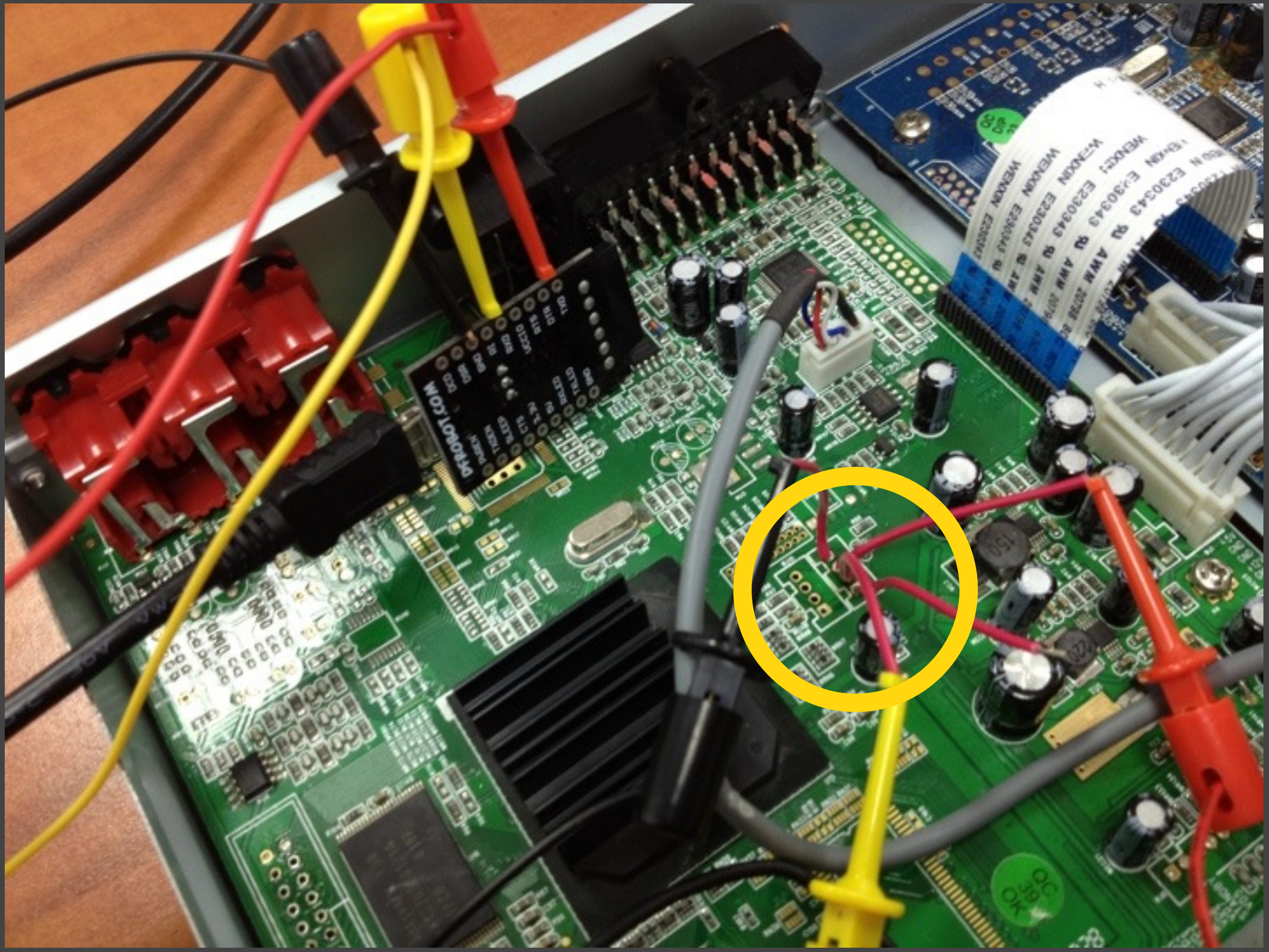
- Many embedded systems use UART as debug output/ console/root shell
  - Exploitee.rs Wiki (formerly GTVHackers), `www.exploitee.rs`
- Converts logic level asynchronous serial to Virtual COM Port
  - → TXD = Transmit data (to target device)
  - ← RXD = Receive data (from target device)
  - ↔ DTR, DSR, RTS, CTS, RI, DCD = Control signals (often unused)
- Easily connects to PC, Mac, Linux w/ suitable drivers
- Ex.: FTDI FT232, CP2102, PL2303, Adafruit FTDI Friend



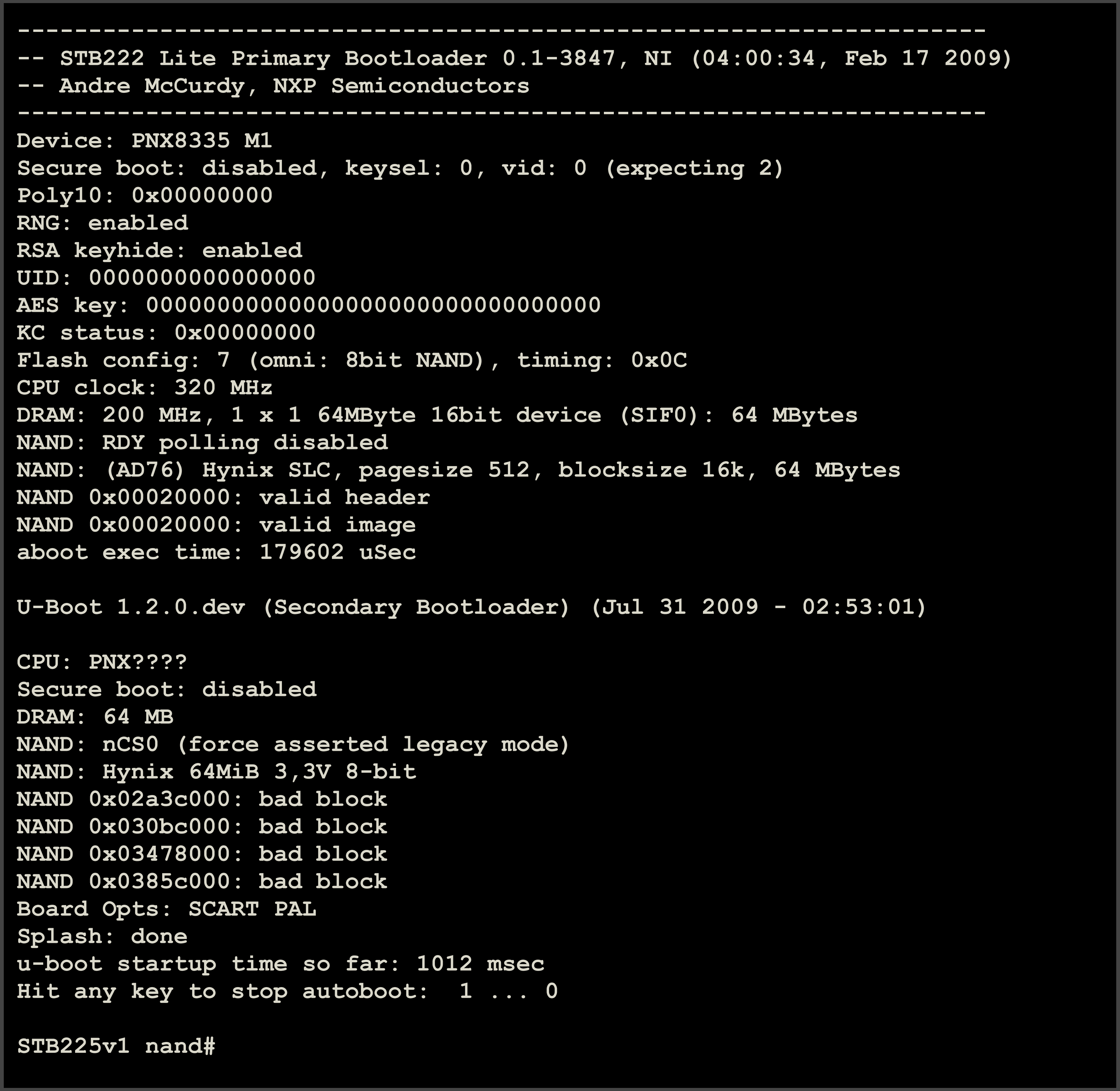


USB-to-Serial Adapter: Example

- Apex STB236 Set Top Box
  - Visually identify connector
  - Oscilloscope to determine baud rate (115.2kbps)
  - USB-to-Serial adapter
  - Bootloader + U-Boot



USB-to-Serial Adapter: Example 2



```
--------------------------------------------------------------------
-- STB222 Lite Primary Bootloader 0.1-3847, NI (04:00:34, Feb 17 2009)
-- Andre McCurdy, NXP Semiconductors
--------------------------------------------------------------------
Device: PNX8335 M1
Secure boot: disabled, keysel: 0, vid: 0 (expecting 2)
Poly10: 0x00000000
RNG: enabled
RSA keyhide: enabled
UID: 0000000000000000
AES key: 00000000000000000000000000000000
KC status: 0x00000000
Flash config: 7 (omni: 8bit NAND), timing: 0x0C
CPU clock: 320 MHz
DRAM: 200 MHz, 1 x 1 64MByte 16bit device (SIF0): 64 MBytes
NAND: RDY polling disabled
NAND: (AD76) Hynix SLC, pagesize 512, blocksize 16k, 64 MBytes
NAND 0x00020000: valid header
NAND 0x00020000: valid image
aboot exec time: 179602 uSec

U-Boot 1.2.0.dev (Secondary Bootloader) (Jul 31 2009 - 02:53:01)

CPU: PNX????
Secure boot: disabled
DRAM: 64 MB
NAND: nCS0 (force asserted legacy mode)
NAND: Hynix 64MiB 3,3V 8-bit
NAND 0x02a3c000: bad block
NAND 0x030bc000: bad block
NAND 0x03478000: bad block
NAND 0x0385c000: bad block
Board Opts: SCART PAL
Splash: done
u-boot startup time so far: 1012 msec
Hit any key to stop autoboot:  1 ... 0

STB225v1 nand#
```

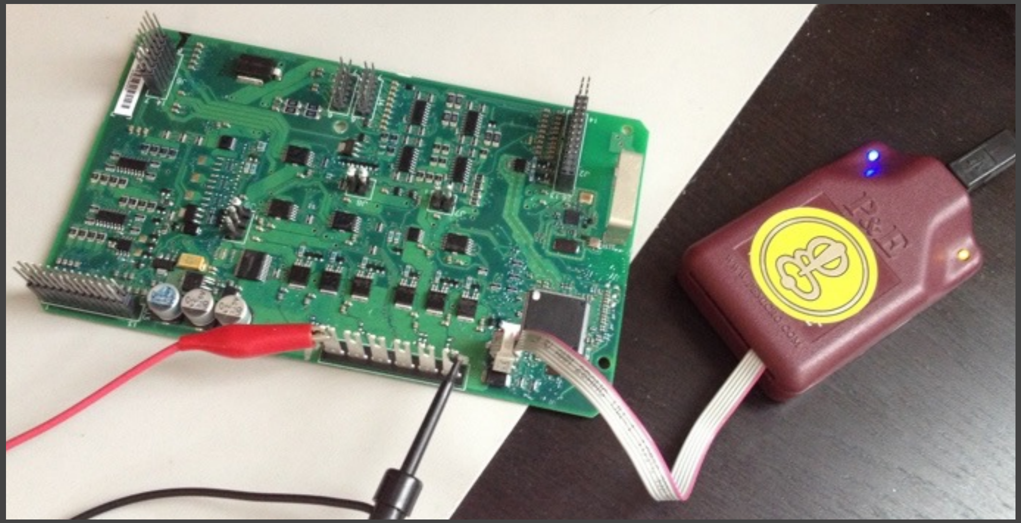
Debug Tools

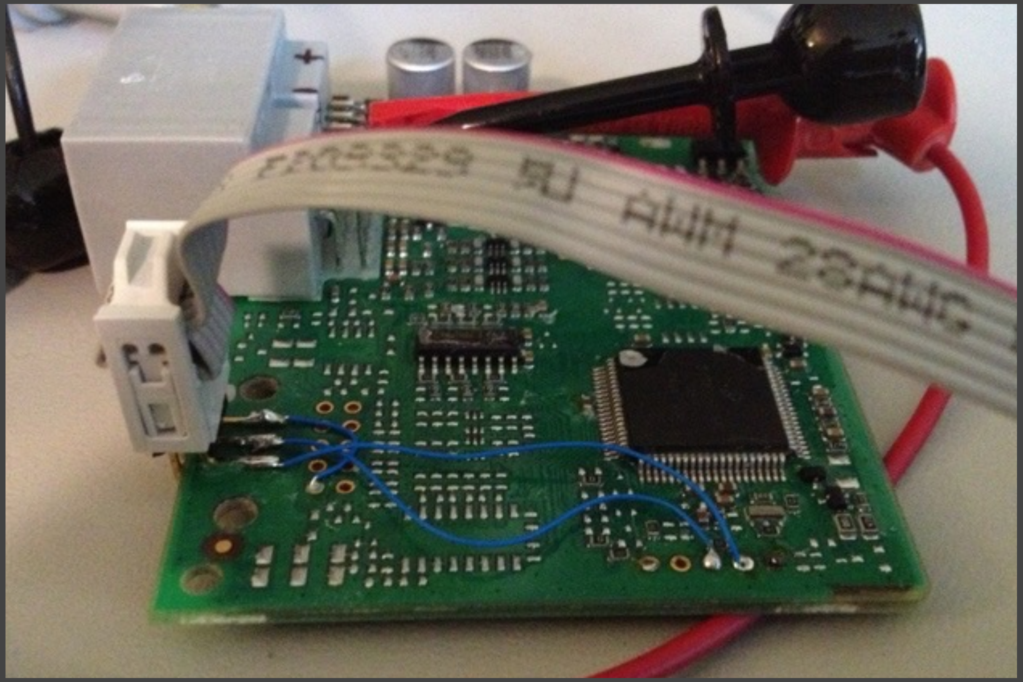
- Off-the-shelf HW tools designed for interaction w/ target device
  - – Can provide chip-level control (single step, access registers)
  - – Extract program code or data
  - – Modify memory contents
  - – Affect device operation on-the-fly
- Either vendor-specific or industry standard (JTAG)
- Many different types available
  - – Ensure tool supports your target architecture
  - – Find out what vendor recommends for legitimate engineers

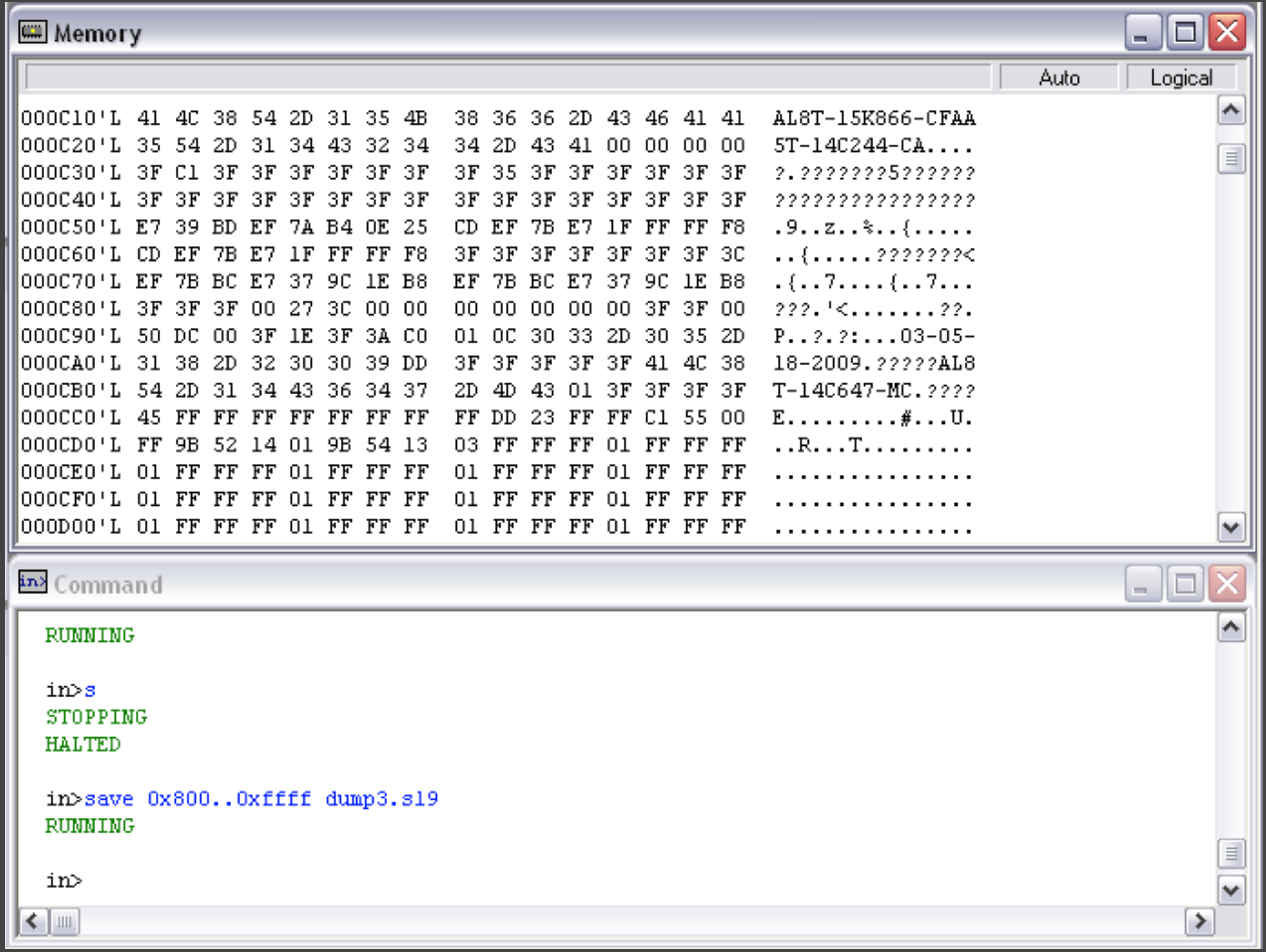
Debug Tools: Example

- Ford Electronic Control Units (ECUs) (2013)
  - For Charlie Miller & Chris Valasek
  - Complete firmware extraction to help understand typical CAN traffic/functionality
  - `http://illmatics.com/car_hacking.pdf`
  - Used standard, off-the-shelf development tools
    - Freescale CodeWarrior for S12(X) v5.1 + P&E Multilink USB Rev. C

Debug Tools: Example 2





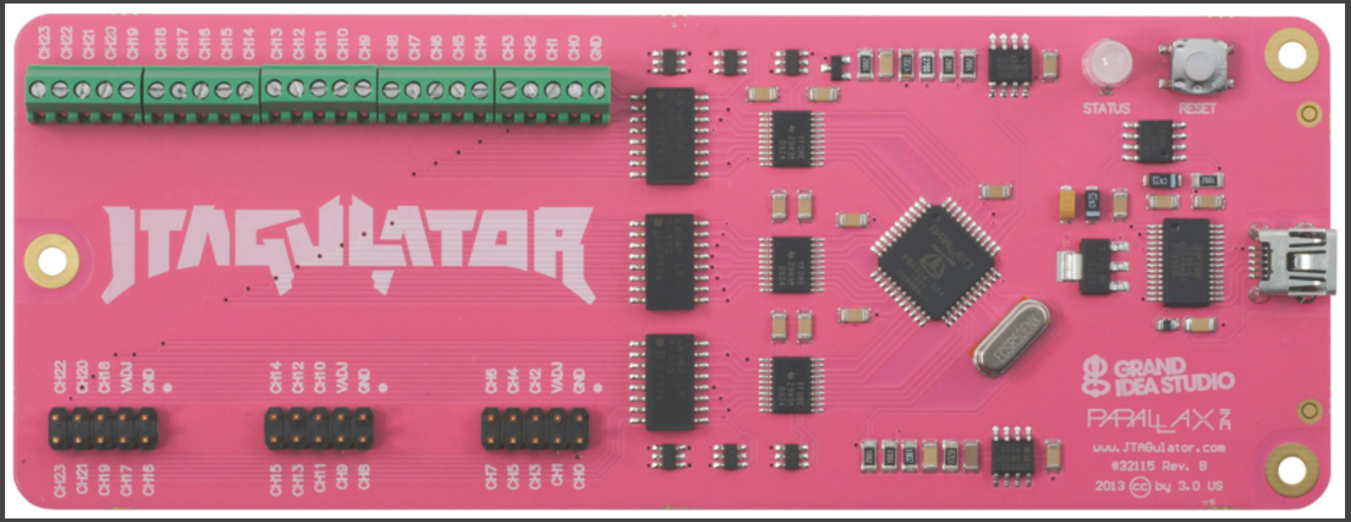


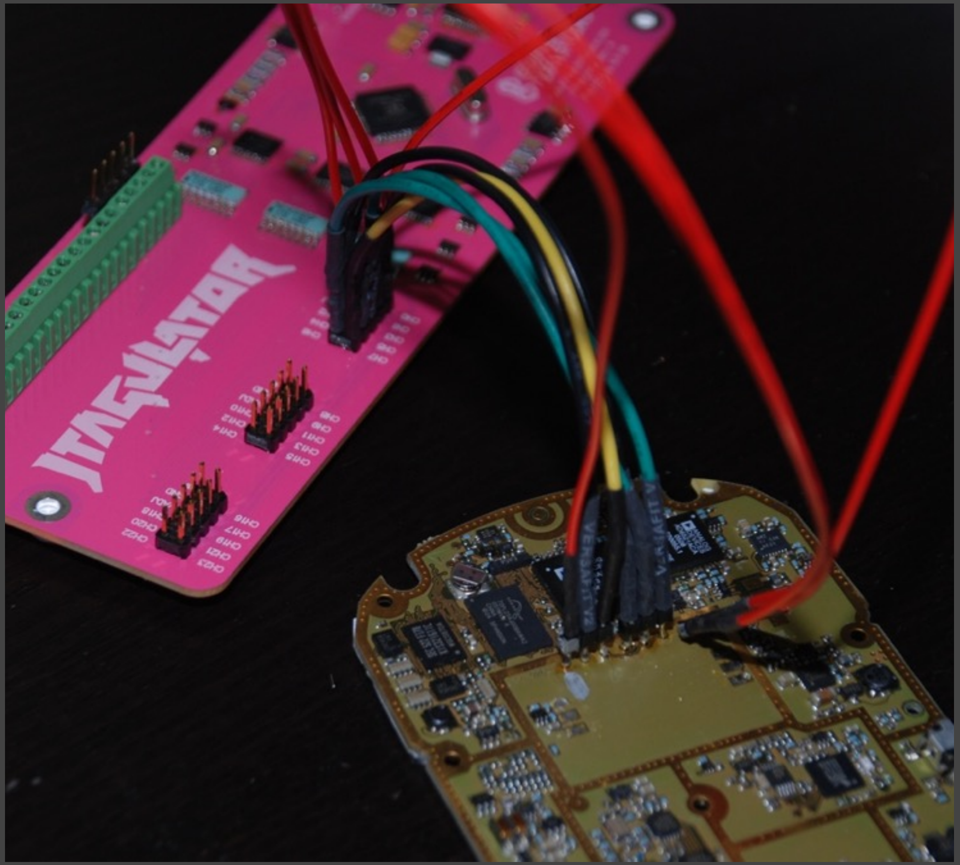
```
000C10'L 41 4C 38 54 2D 31 35 4B  38 36 36 2D 43 46 41 41  AL8T-15K866-CFAA
000C20'L 35 54 2D 31 34 43 32 34  34 2D 43 41 00 00 00 00  5T-14C244-CA....
000C30'L 3F C1 3F 3F 3F 3F 3F 3F  3F 35 3F 3F 3F 3F 3F 3F  ?.???????5??????
000C40'L 3F 3F 3F 3F 3F 3F 3F 3F  3F 3F 3F 3F 3F 3F 3F 3F  ????????????????
000C50'L E7 39 BD EF 7A B4 0E 25  CD EF 7B E7 1F FF FF F8  .9..z..%..{.....
000C60'L CD EF 7B E7 1F FF FF F8  3F 3F 3F 3F 3F 3F 3F 3C  ..{.....???????<
000C70'L EF 7B BC E7 37 9C 1E B8  EF 7B BC E7 37 9C 1E B8  .{..7....{..7...
000C80'L 3F 3F 3F 00 27 3C 00 00  00 00 00 00 00 3F 3F 00  ???.'<.......??.
000C90'L 50 DC 00 3F 1E 3F 3A C0  01 0C 30 33 2D 30 35 2D  P..?.?:...03-05-
000CA0'L 31 38 2D 32 30 30 39 DD  3F 3F 3F 3F 3F 41 4C 38  18-2009.?????AL8
000CB0'L 54 2D 31 34 43 36 34 37  2D 4D 43 01 3F 3F 3F 3F  T-14C647-MC.????
000CC0'L 45 FF FF FF FF FF FF FF  FF DD 23 FF FF C1 55 00  E.........#...U.
000CD0'L FF 9B 52 14 01 9B 54 13  03 FF FF FF 01 FF FF FF  ..R...T.........
000CE0'L 01 FF FF FF 01 FF FF FF  01 FF FF FF 01 FF FF FF  ................
000CF0'L 01 FF FF FF 01 FF FF FF  01 FF FF FF 01 FF FF FF  ................
000D00'L 01 FF FF FF 01 FF FF FF  01 FF FF FF 01 FF FF FF  ................
```

```
RUNNING

in>s
STOPPING
HALTED

in>save 0x800..0xffff dump3.s19
RUNNING

in>
```

Debug Tools: JTAGulator

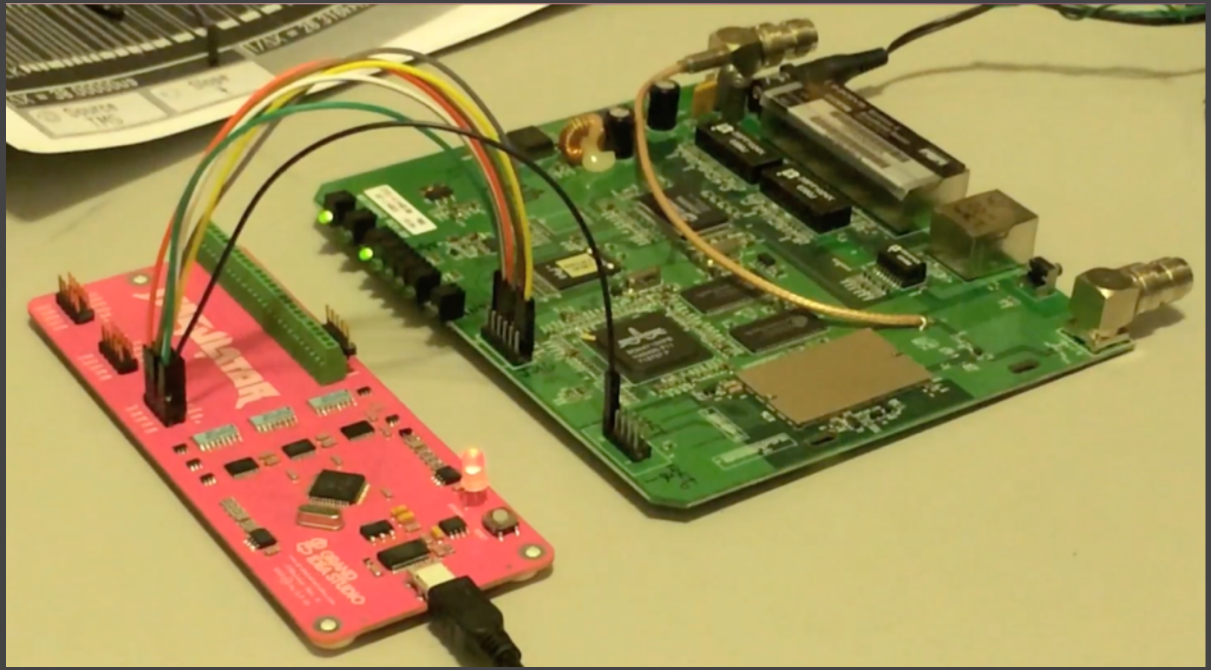
- Open source tool to assist with discovery of on-chip program/ debug interfaces
- Currently detects JTAG & UART/asynchronous serial
- Supports up to 24 connections to unknown points on target circuit board, adjustable target voltage (1.2V-3.3V), input protection, firmware upgradable
- `www.jtagulator.com`

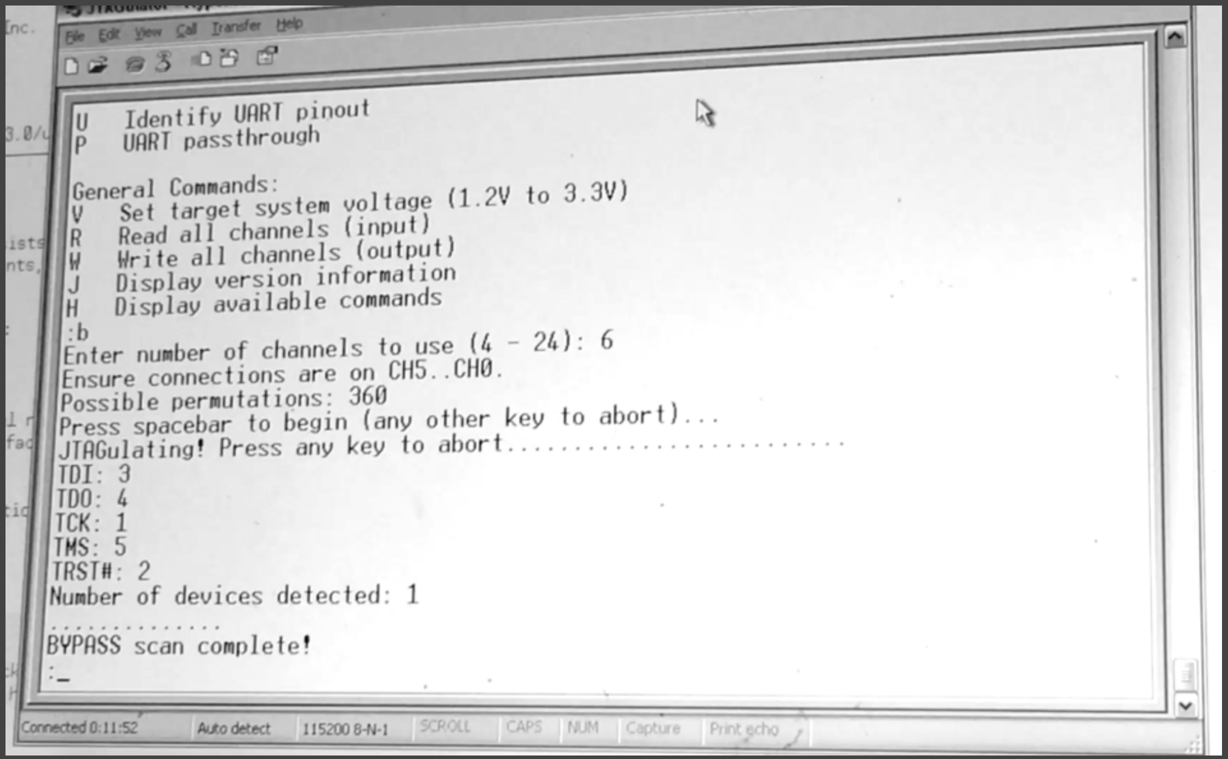




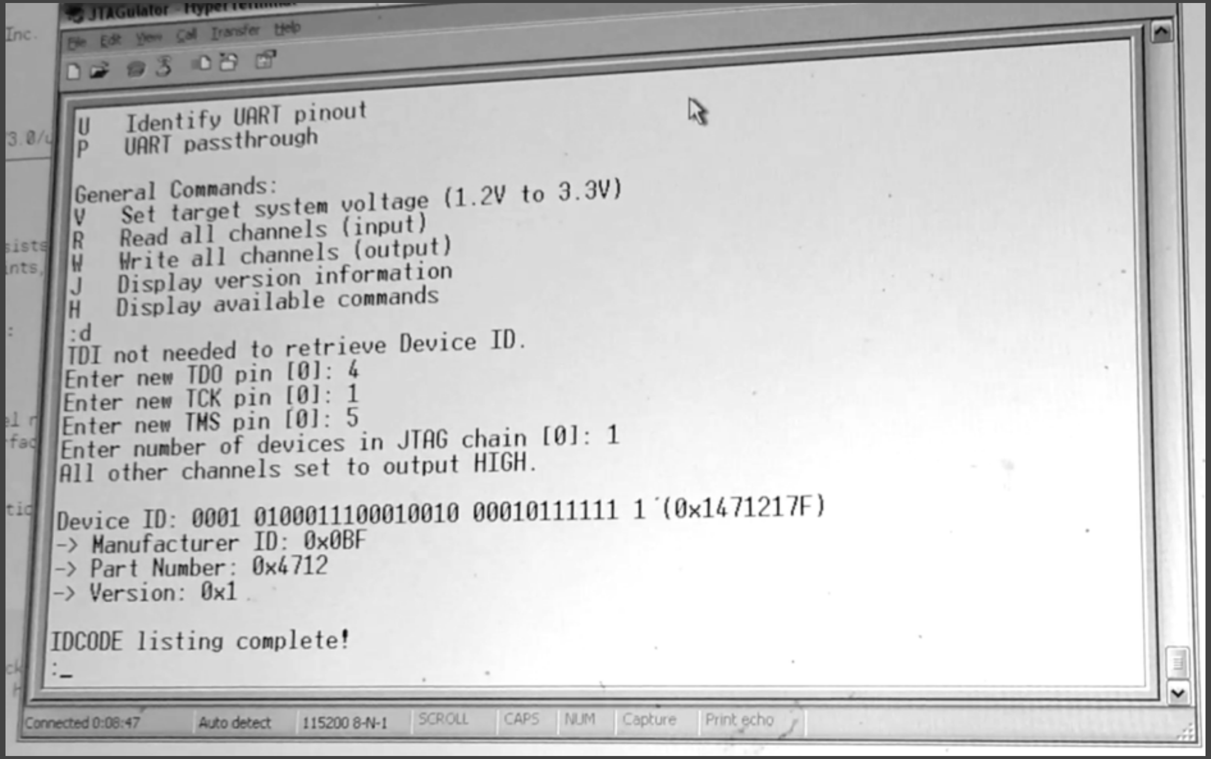
Debug Tools: JTAGulator Example

- Linksys WRT54G v2
  - Broadcom BCM4712
  - IDCODE = 0x1471217F





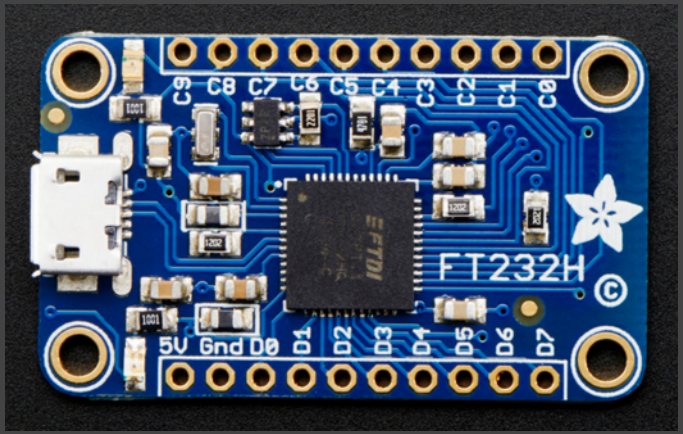
```
U   Identify UART pinout
P   UART passthrough

General Commands:
V   Set target system voltage (1.2V to 3.3V)
R   Read all channels (input)
W   Write all channels (output)
J   Display version information
H   Display available commands
:b
Enter number of channels to use (4 - 24): 6
Ensure connections are on CH5..CH0.
Possible permutations: 360
Press spacebar to begin (any other key to abort)...
JTAGulating! Press any key to abort.........................
TDI: 3
TDO: 4
TCK: 1
TMS: 5
TRST#: 2
Number of devices detected: 1
..............
BYPASS scan complete!
:_
```

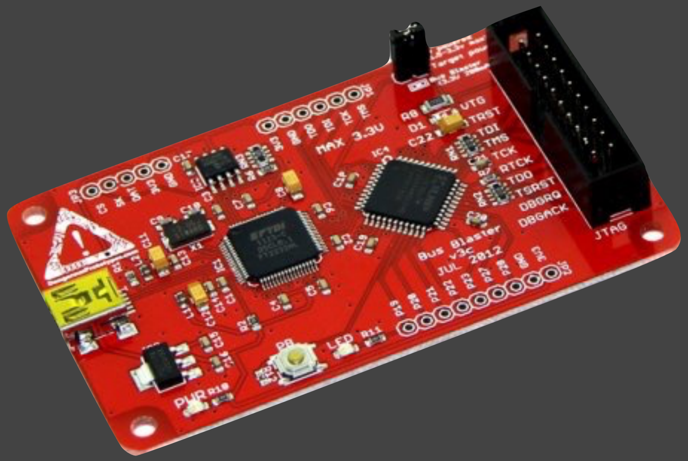


```
U   Identify UART pinout
P   UART passthrough

General Commands:
V   Set target system voltage (1.2V to 3.3V)
R   Read all channels (input)
W   Write all channels (output)
J   Display version information
H   Display available commands
:d
TDI not needed to retrieve Device ID.
Enter new TDO pin [0]: 4
Enter new TCK pin [0]: 1
Enter new TMS pin [0]: 5
Enter number of devices in JTAG chain [0]: 1
All other channels set to output HIGH.

Device ID: 0001 0100011100010010 00010111111 1 (0x1471217F)
-> Manufacturer ID: 0x0BF
-> Part Number: 0x4712
-> Version: 0x1

IDCODE listing complete!
:_
```

Debug Tools: JTAG

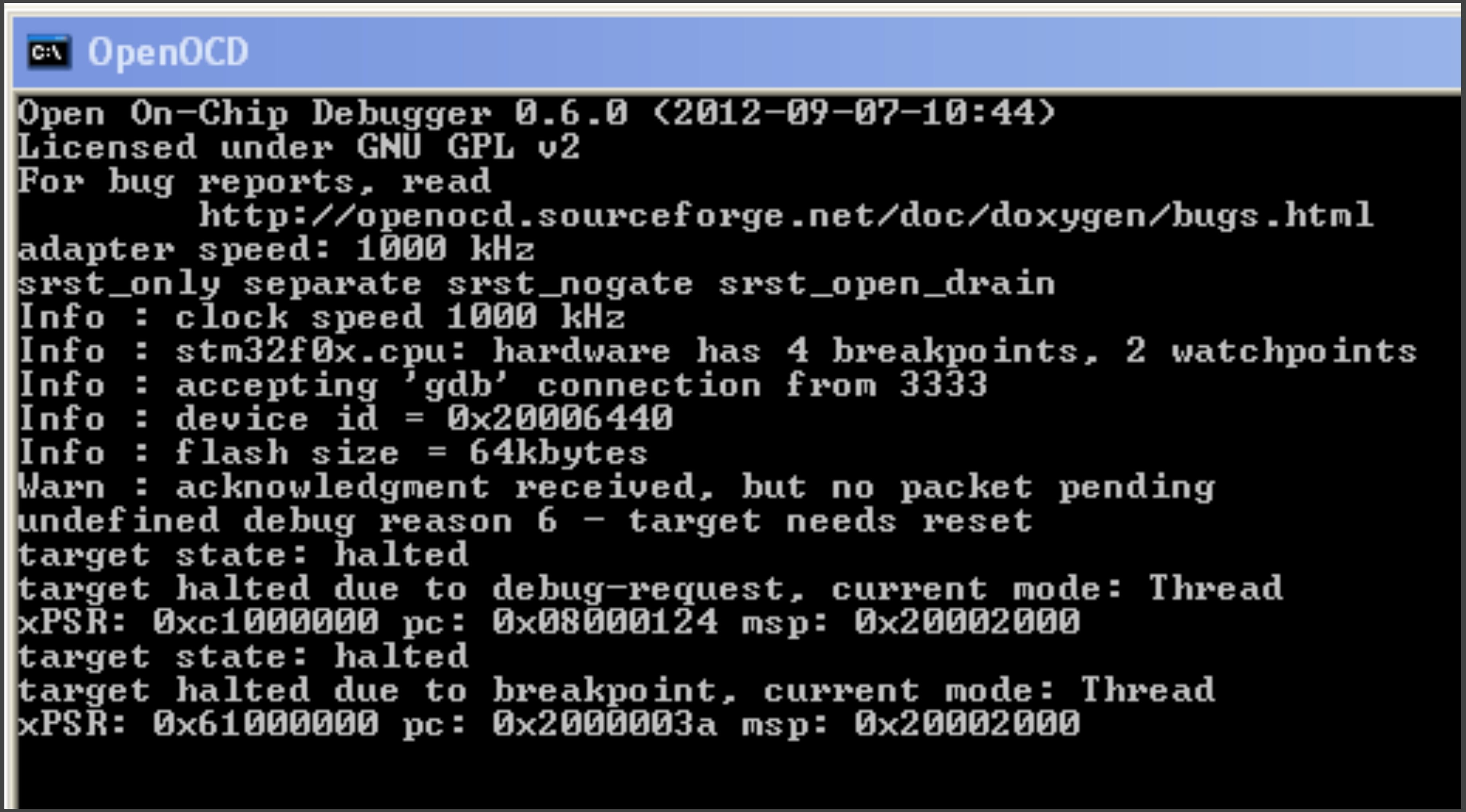
- Bus Blaster (open source)
  - `http://dangerousprototypes.com/docs/Bus_Blaster`
- FT232H Breakout Board
  - `www.adafruit.com/product/2264`
- SEGGER J-Link
  - `www.segger.com/debug-probes.html`
- H-JTAG
  - `www.hjtag.com/en`
- RIFF Box
  - `www.riffbox.org`
- Many Others
  - `http://openocd.sourceforge.net/doc/html/Debug-Adapter-Hardware.html`





Debug Tools: JTAG (SW)

- Open On-Chip Debugger (OpenOCD)
  - `http://openocd.sourceforge.net`
- UrJTAG (Universal JTAG Library)
  - `www.urjtag.org`



```
Open On-Chip Debugger 0.6.0 (2012-09-07-10:44)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.sourceforge.net/doc/doxygen/bugs.html
adapter speed: 1000 kHz
srst_only separate srst_nogate srst_open_drain
Info : clock speed 1000 kHz
Info : stm32f0x.cpu: hardware has 4 breakpoints, 2 watchpoints
Info : accepting 'gdb' connection from 3333
Info : device id = 0x20006440
Info : flash size = 64kbytes
Warn : acknowledgment received, but no packet pending
undefined debug reason 6 - target needs reset
target state: halted
target halted due to debug-request, current mode: Thread
xPSR: 0xc1000000 pc: 0x08000124 msp: 0x20002000
target state: halted
target halted due to breakpoint, current mode: Thread
xPSR: 0x61000000 pc: 0x2000003a msp: 0x20002000
```

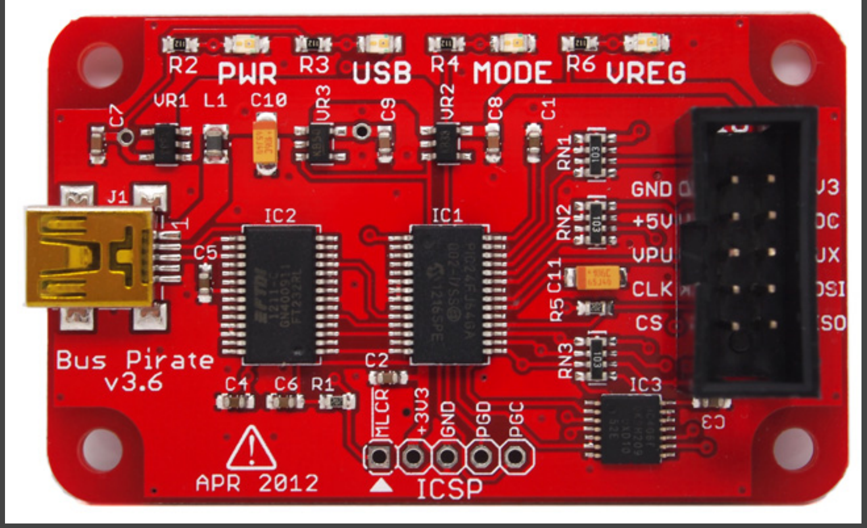
Debug Tools: JTAG Example

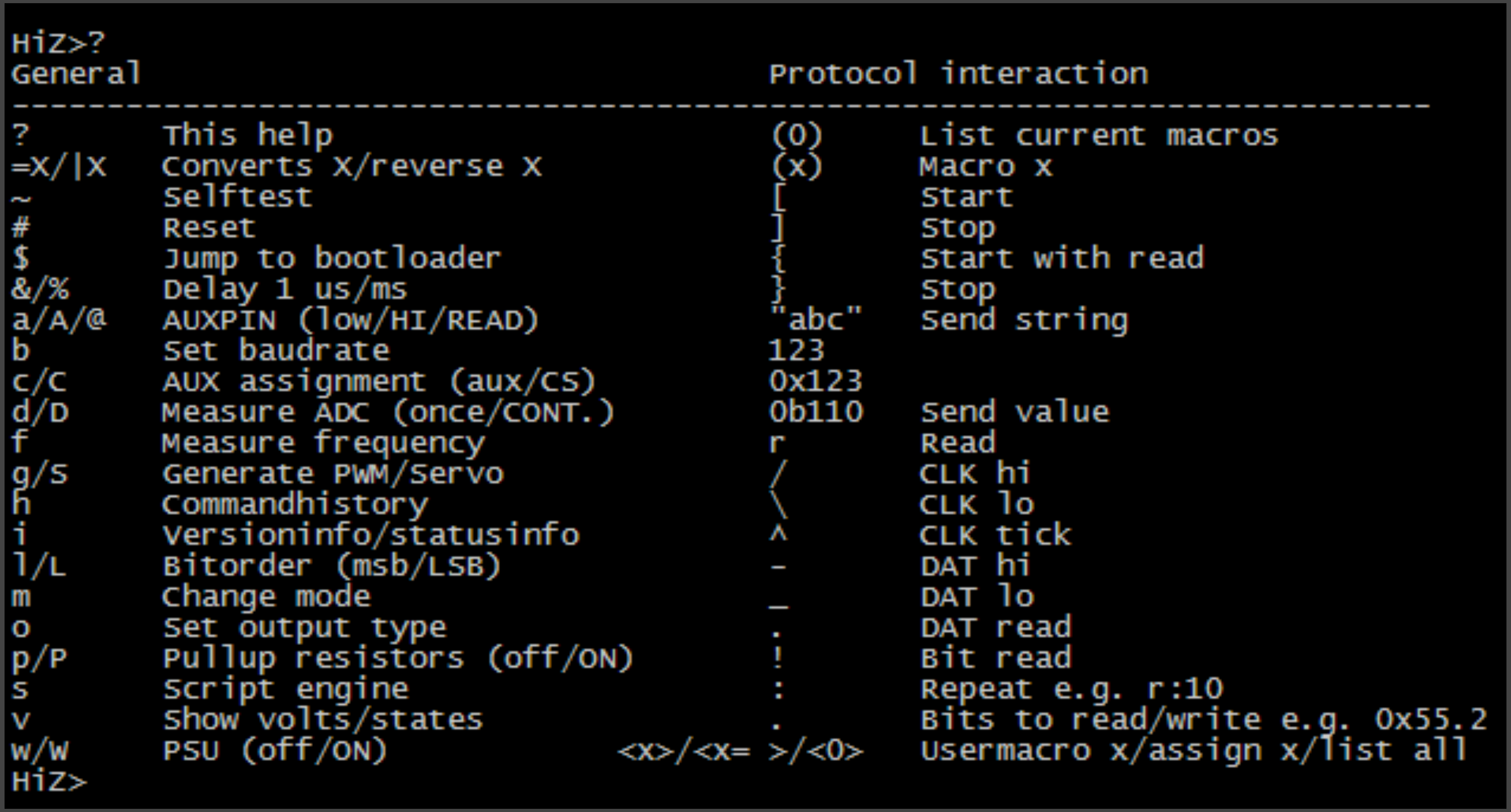
- JTAG to Root, 5 Ways, Joe FitzPatrick & Matt King, BSides PDX 2015
  - `https://github.com/syncsrc/jtagsploitation`

1. Access non-volatile memory via boundary scan
2. Scrape memory for offline analysis
3. Patch boot arguments
4. Patch kernel
5. Patch a process

Bus Pirate

- Open source tool to interface w/ serial devices
  - SPI, I2C, 1-Wire, LCD, MIDI, MCU/FPGA programming, bit bang
- Basic logic analyzer/digital decoding functionality (slow)
- **`http://dangerousprototypes.com/docs/Bus_Pirate`**

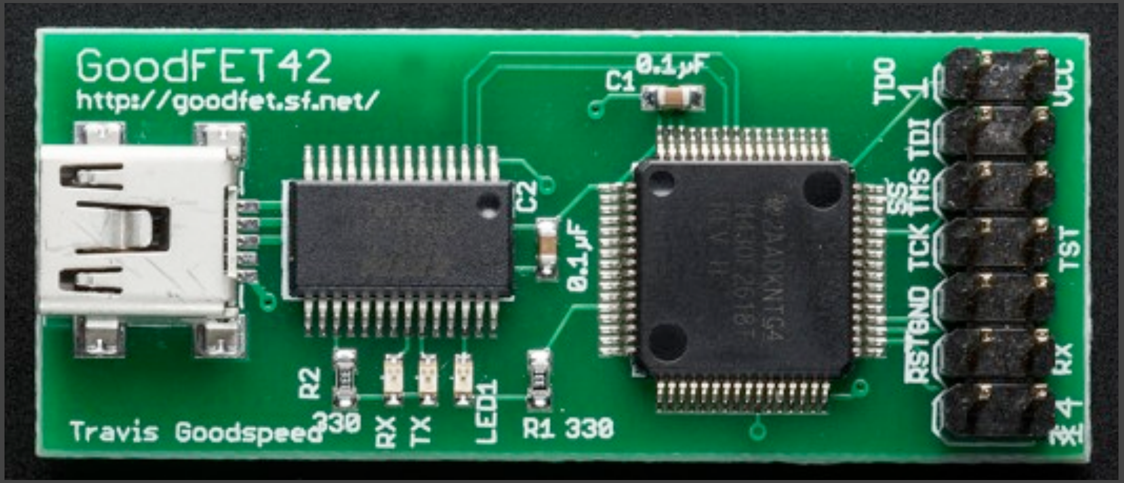




```
HiZ>?
General                                         Protocol interaction
---------------------------------------------------------------------------------------------
?         This help                             (0)       List current macros
=X/|X     Converts X/reverse X                  (x)       Macro x
~         Selftest                              [         Start
#         Reset                                 ]         Stop
$         Jump to bootloader                    {         Start with read
&/%       Delay 1 us/ms                         }         Stop
a/A/@     AUXPIN (low/HI/READ)                  "abc"     Send string
b         Set baudrate                          123
c/C       AUX assignment (aux/CS)               0x123
d/D       Measure ADC (once/CONT.)              0b110     Send value
f         Measure frequency                     r         Read
g/S       Generate PWM/Servo                    /         CLK hi
h         Commandhistory                        \         CLK lo
i         Versioninfo/statusinfo                ^         CLK tick
l/L       Bitorder (msb/LSB)                    -         DAT hi
m         Change mode                           _         DAT lo
o         Set output type                       .         DAT read
p/P       Pullup resistors (off/ON)             !         Bit read
s         Script engine                         :         Repeat e.g. r:10
v         Show volts/states                     .         Bits to read/write e.g. 0x55.2
w/W       PSU (off/ON)                <x>/<x= >/<0>       Usermacro x/assign x/list all
HiZ>
```

GoodFET

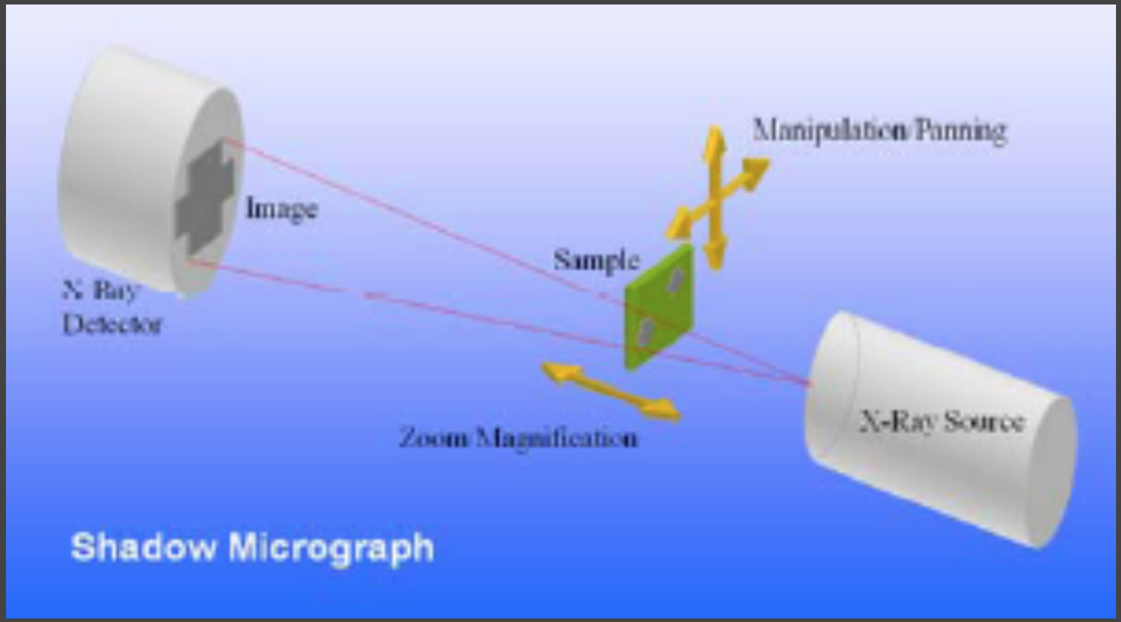
- Travis Goodspeed
- Open source tool for interfacing to/hacking devices
- Different FW and Python scripts for different functionality
  - Ex.: JTAG, SPI, I2C, AVR, PIC, Chipcon/Nordic/Atmel RF
- `http://goodfet.sourceforge.net`



Imaging

X-Ray (2D)

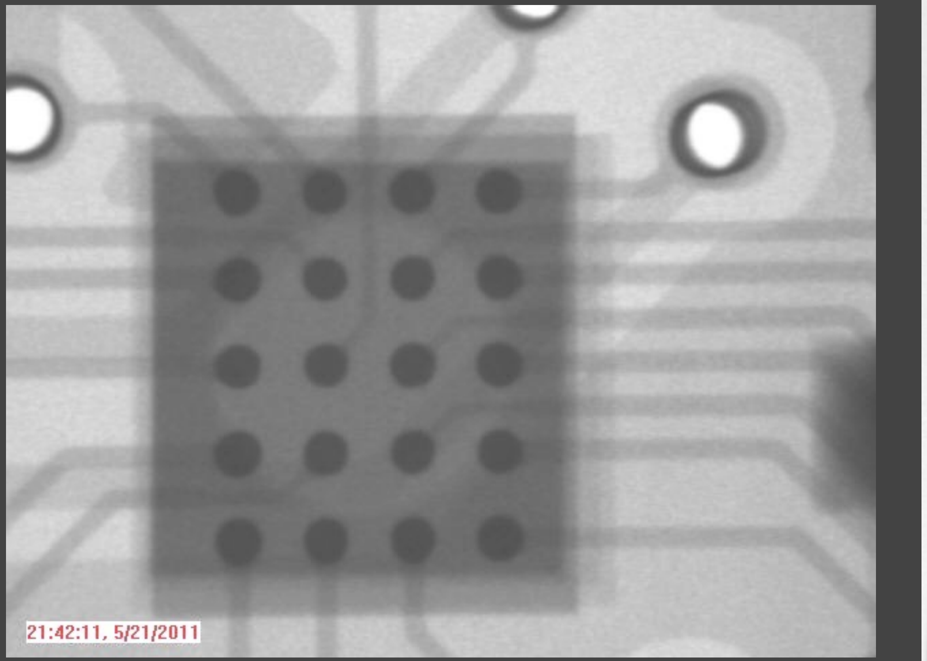
- X-rays passed through target and received on detector
  - All materials absorb radiation differently depending on density, atomic number, and thickness
- Provides a composite image of all layers in target

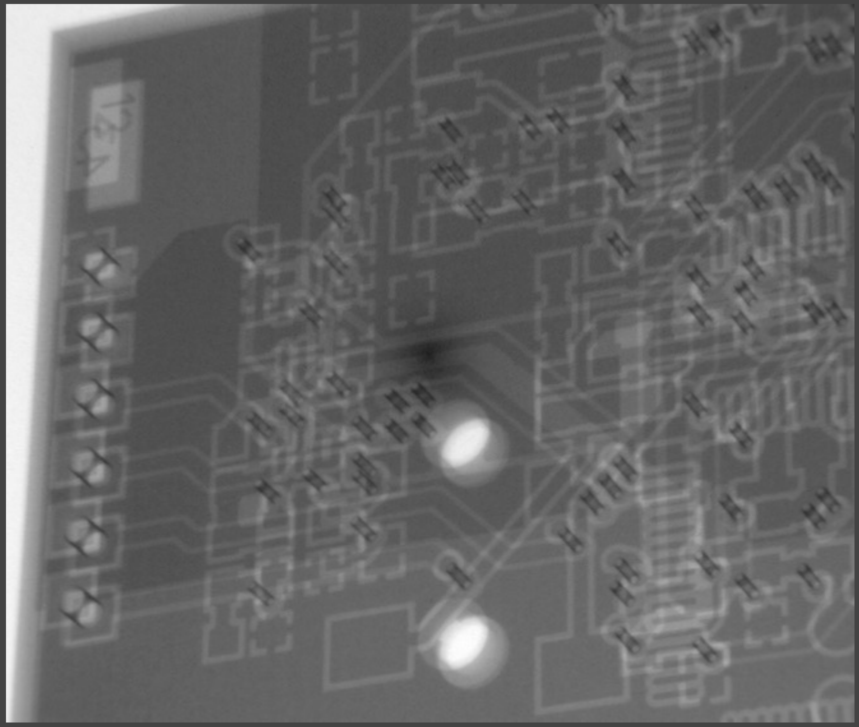


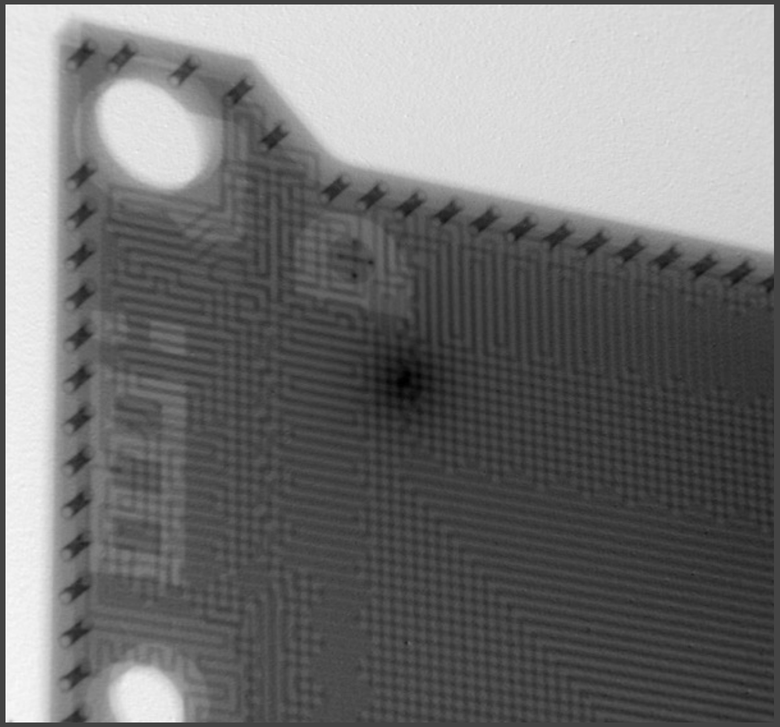
http://datest.com/resources-boardtestmeth-primer2d3d.php

X-Ray (2D) 2

- Typically used during PCB assembly (component placement/ solder quality) or failure analysis (troubleshooting defective features)
- We can use it for general PCB inspection and examining through epoxy encapsulation
  - Can get clues of PCB fabrication techniques, component location, layer count, hidden/embedded features

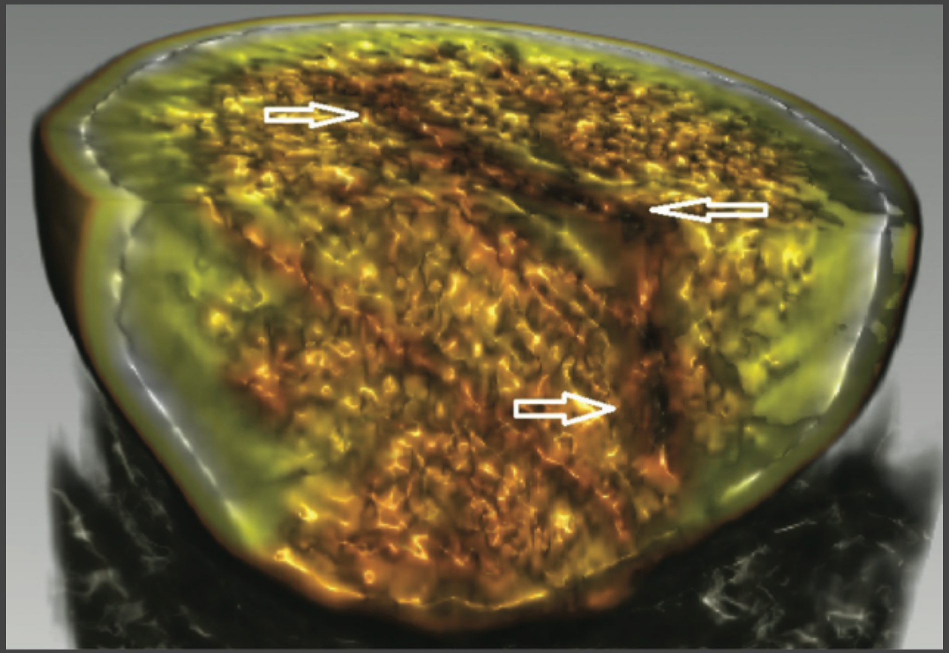


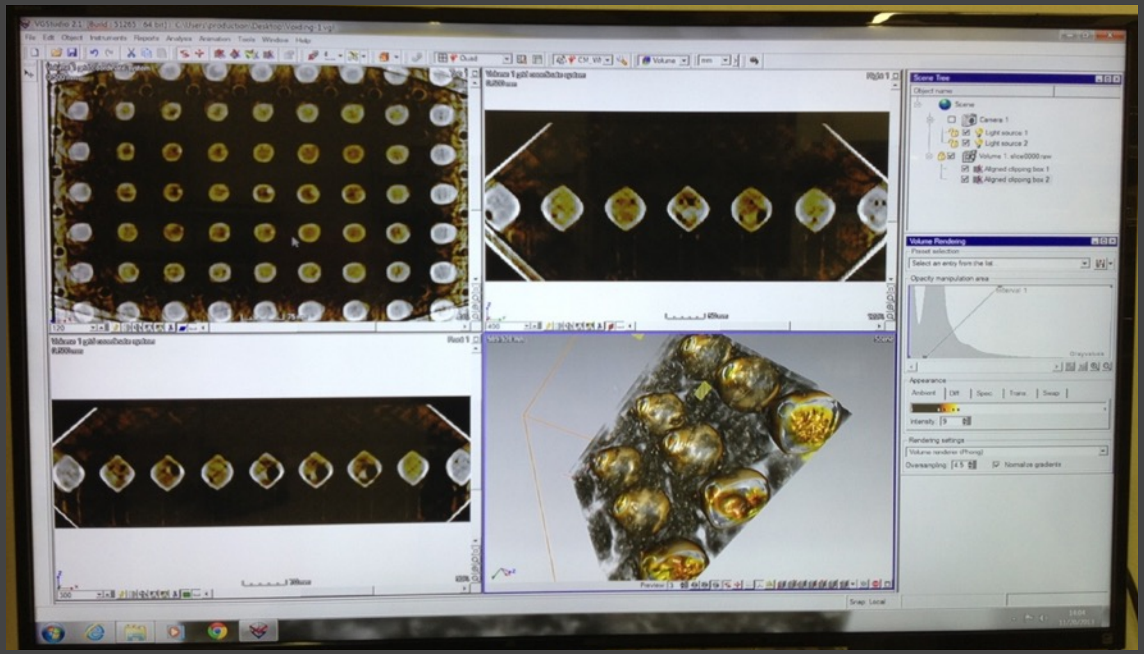




X-Ray (3D/CT)

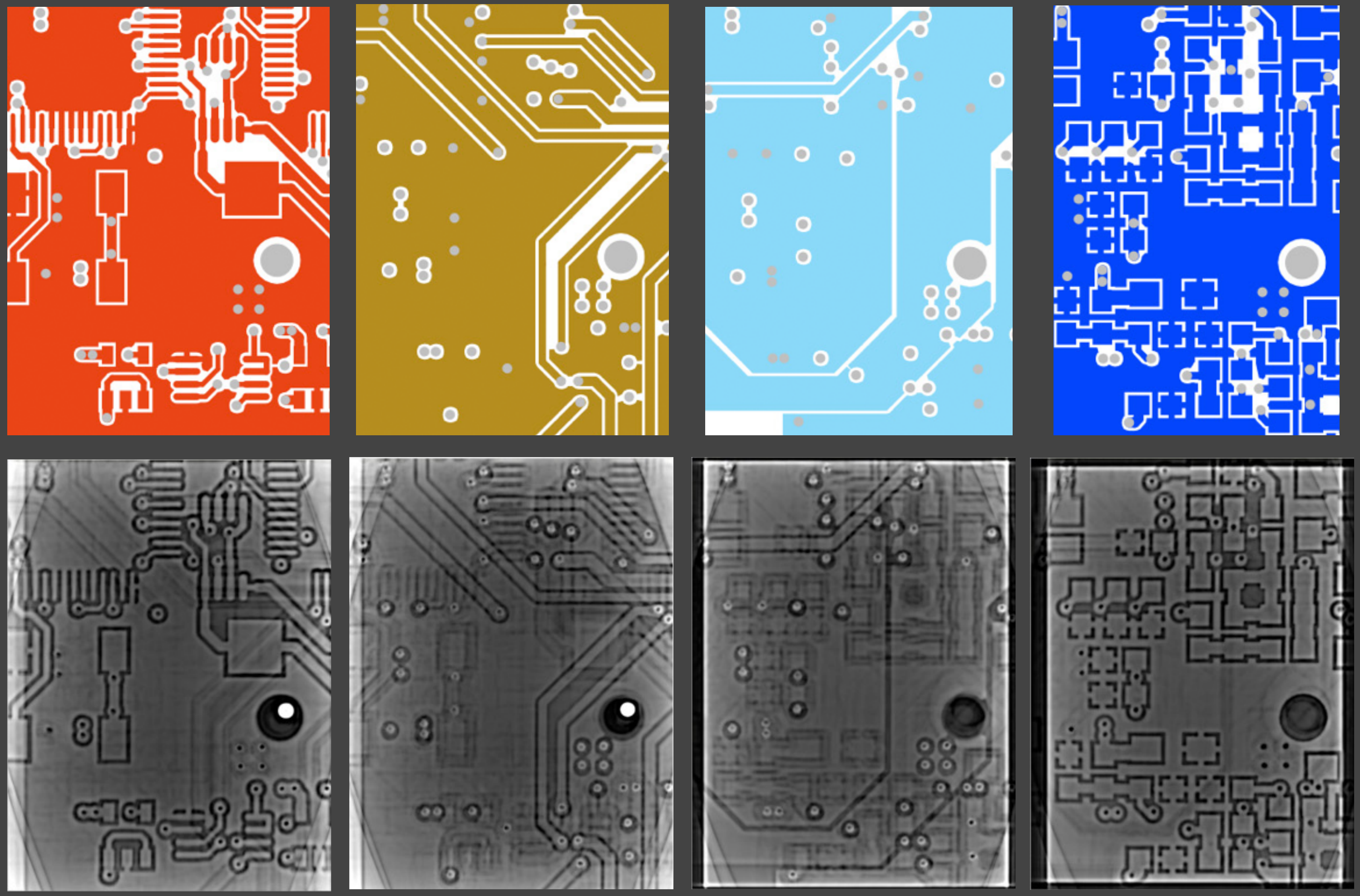
- Computed Tomography (CT)
  - A series of 2D X-ray images post-processed to create cross-sectional slices of the target
  - X-ray beam rotated 360° in a single axis around the target
  - Post-processing results in 2D slices that can be viewed in any plane (X, Y, Z)
  - Can be manipulated with 3D modeling software





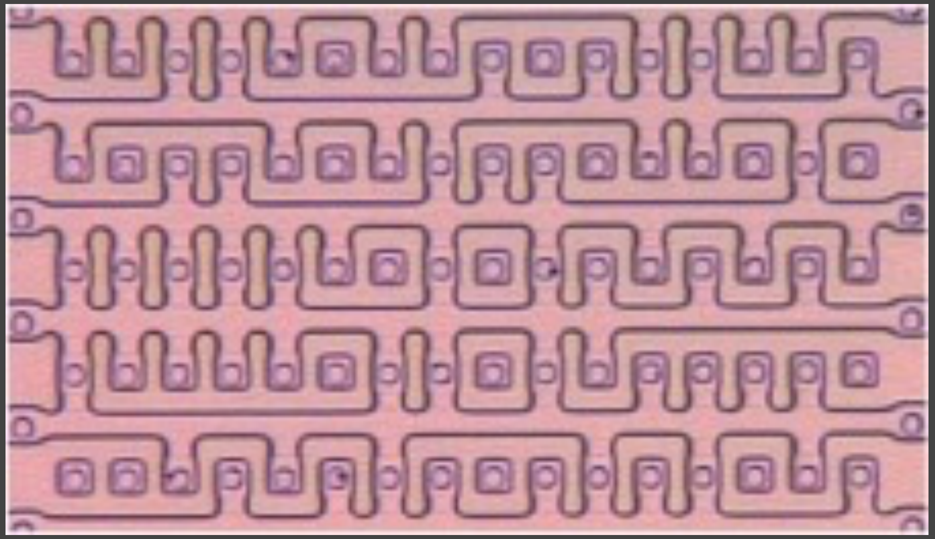
X-Ray (3D/CT) 2

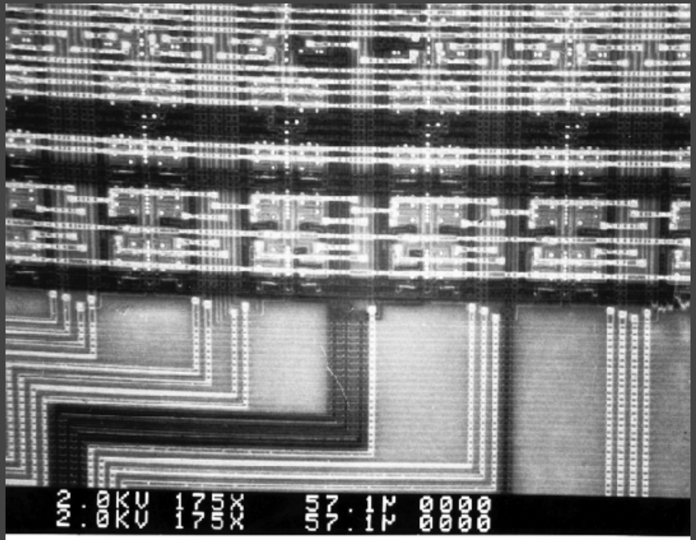
- Typically used for complex inspection and failure analysis of PCBs, component packaging, solder ball/joint quality
- We can use it to extract individual layers of a PCB
  - Results may vary based on layer count, inter-layer thickness, copper weight, substrate composition



Scanning Electron Microscope

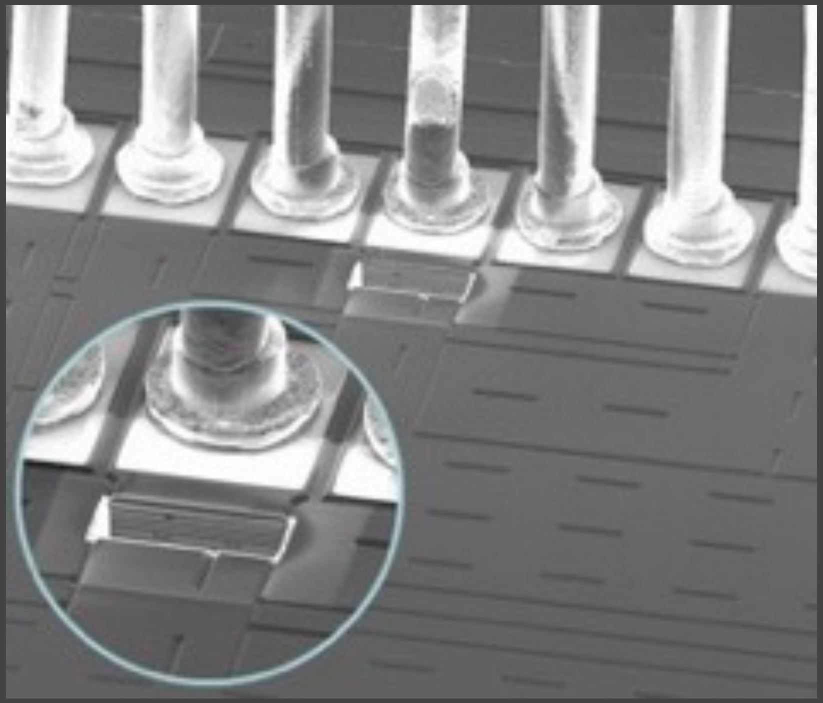
- Uses electrons instead of light to form an image
  - Wide range of magnifications, better quality than optical microscope
- Provides an entire chip-level and gate-level view of the device
  - Will need to remove other metal or glass layers before getting access to gate structures (polysilicon)
- Voltage contrast microscopy
  - Gate charges and voltage levels shown as brightness variations
  - Useful for failure analysis/comparisons and signal/bus monitoring





FIB (Focused Ion Beam)

- Send a focused stream of ions onto the surface of the chip
- Beam current/velocity and optional use of gas/vapor changes the function:
  - Imaging
  - Cutting
  - Deposition



What Now?

- Create a hardware hacking lab (if you haven't already)
- Keep an eye out for new tools by hackers and industry
- Collaborate with others who may have complementary skills/tools
- Use these tools to validate your product's security or to better understand attack techniques

The End.