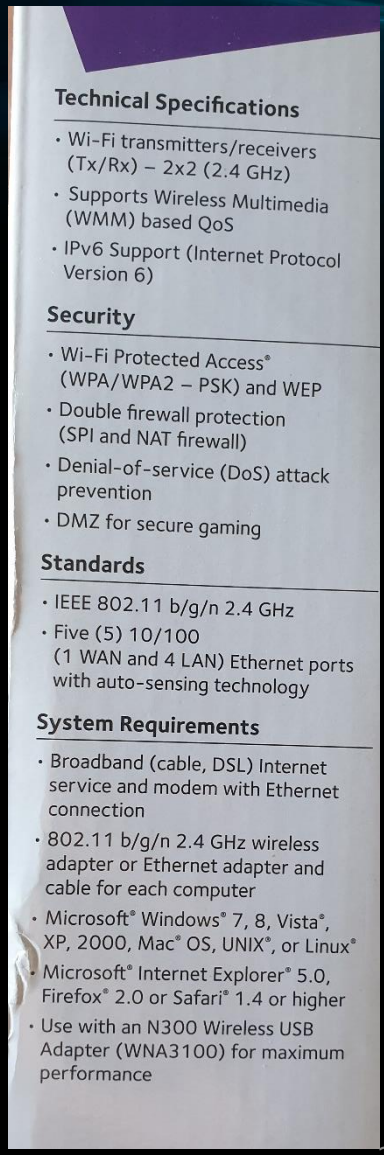
HARDWARE HACKING EXPERIMENTS

Extracting Firmware from Embedded Device

Jérémy Brun-Nouvion - 2020

The Target

- Device: Netgear N300 Wireless Router
- Model No: WNR2000v4



Technical Specifications

- Wi-Fi transmitters/receivers (Tx/Rx) – 2x2 (2.4 GHz)
- Supports Wireless Multimedia (WMM) based QoS
- IPv6 Support (Internet Protocol Version 6)

Security

- Wi-Fi Protected Access® (WPA/WPA2 – PSK) and WEP
- Double firewall protection (SPI and NAT firewall)
- Denial-of-service (DoS) attack prevention
- DMZ for secure gaming

Standards

- IEEE 802.11 b/g/n 2.4 GHz
- Five (5) 10/100 (1 WAN and 4 LAN) Ethernet ports with auto-sensing technology

System Requirements

- Broadband (cable, DSL) Internet service and modem with Ethernet connection
- 802.11 b/g/n 2.4 GHz wireless adapter or Ethernet adapter and cable for each computer
- Microsoft® Windows® 7, 8, Vista®, XP, 2000, Mac® OS, UNIX®, or Linux®
- Microsoft® Internet Explorer® 5.0, Firefox® 2.0 or Safari® 1.4 or higher
- Use with an N300 Wireless USB Adapter (WNA3100) for maximum performance

0> TOOLBOX

Tools to open devices

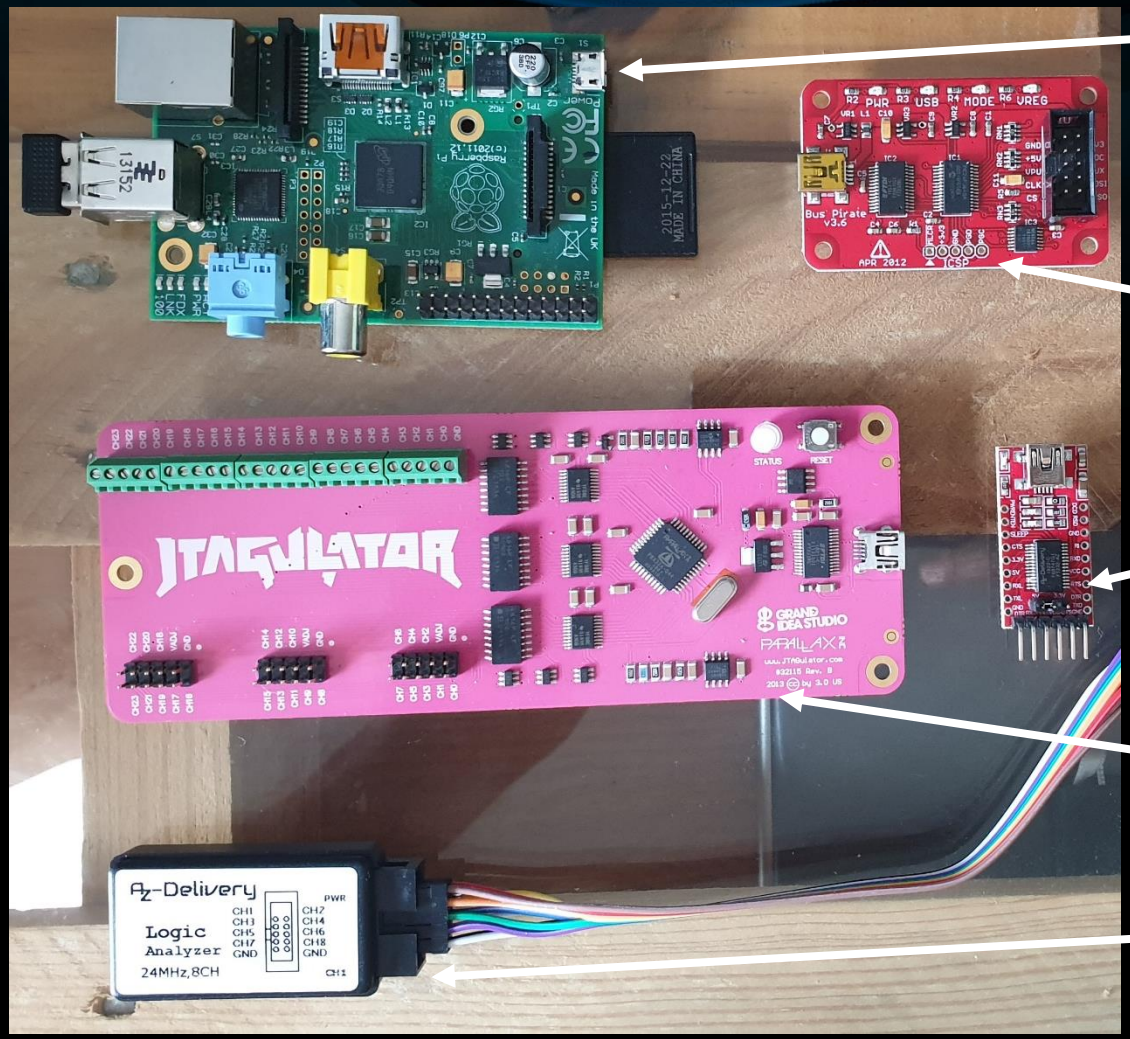
Multimeter - PicoScope

- Multimeter:
  - Very useful feature: Continuity test with beep (detect GND & Vcc pins easily)
  - Measure Voltage:
    - High constant (around 3.3V or 5V) may indicate Vcc
    - Voltage fluctuation may indicate data transmission
- PicoScope =
  - USB PC Oscilloscope
  - Can be used to find points on PCB that are emitting data (eg. Tx pin of UART)

Physical connection tools

- Soldering iron
- Pin headers (to solder to PCB pads)
- Jump wires (m/m, f/m, f/f)
- Chip clips (for 8-pin & 16-pin Flash/EEPROM)

Hardware



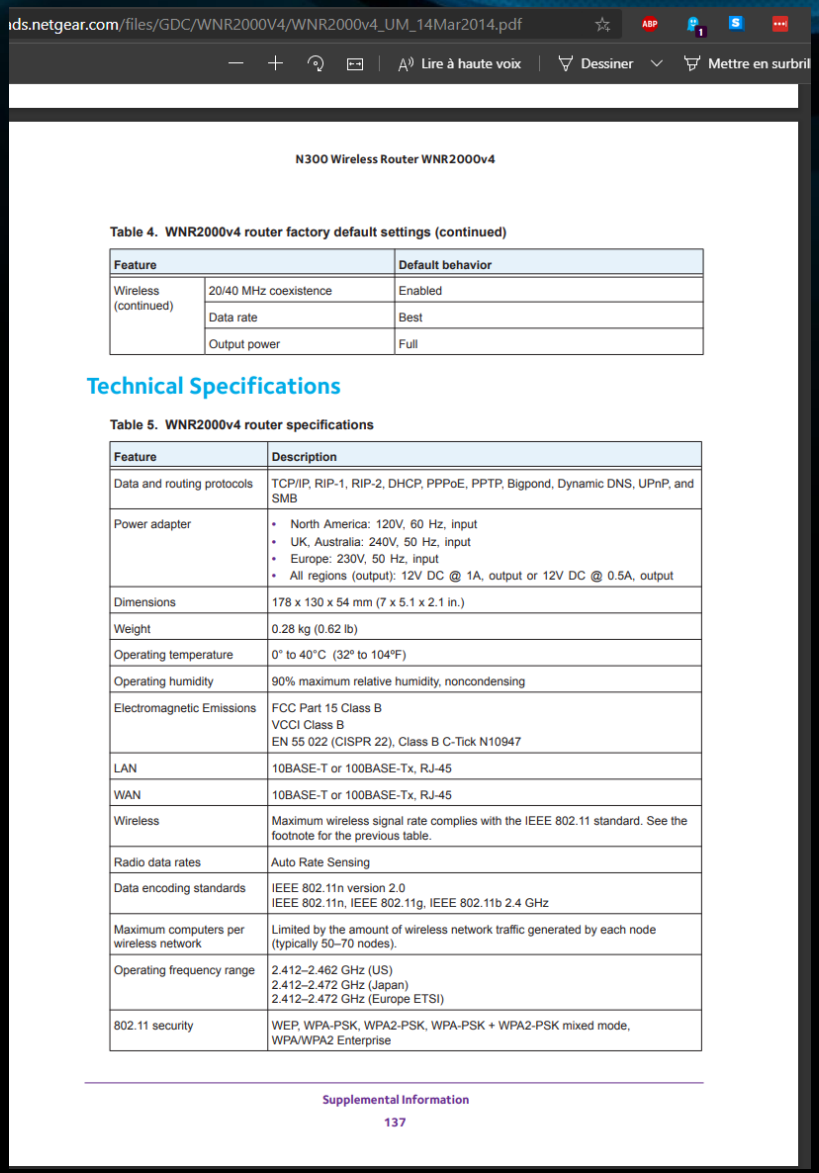
- Raspberry Pi
- Bus Pirate (v3.6): Universal bus interface compatible with multiple protocols (I²C, SPI, JTAG, UART…)
- UART to USB adapter
- JTAGulator: Useful to identify JTAG pins
- Logic Analyzer

Softwares

- Terminal emulator: screen/minicom/putty
- PicoScope software https://www.picotech.com/downloads
- Salae Logic Analyzer https://www.saleae.com/downloads/
- OpenOCD (used to interact with device via JTAG) http://openocd.org/
- Flashrom (identify, read, write flash memory chips) https://www.flashrom.org/
- Binwalk (firmware analysis tool) https://github.com/ReFirmLabs/binwalk

1> RECON

Manual / Online Public Information



ds.netgear.com/files/GDC/WNR2000V4/WNR2000v4_UM_14Mar2014.pdf

Lire à haute voix | Dessiner | Mettre en surbril

N300 Wireless Router WNR2000v4

Table 4. WNR2000v4 router factory default settings (continued)

Feature		Default behavior
Wireless (continued)	20/40 MHz coexistence	Enabled
	Data rate	Best
	Output power	Full

Technical Specifications

Table 5. WNR2000v4 router specifications

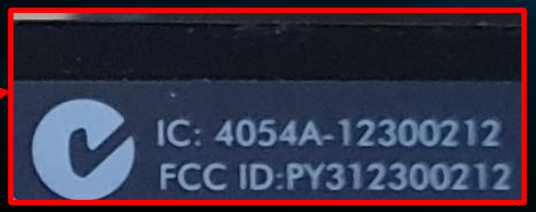
Feature	Description
Data and routing protocols	TCP/IP, RIP-1, RIP-2, DHCP, PPPoE, PPTP, Bigpond, Dynamic DNS, UPnP, and SMB
Power adapter	• North America: 120V, 60 Hz, input • UK, Australia: 240V, 50 Hz, input • Europe: 230V, 50 Hz, input • All regions (output): 12V DC @ 1A, output or 12V DC @ 0.5A, output
Dimensions	178 x 130 x 54 mm (7 x 5.1 x 2.1 in.)
Weight	0.28 kg (0.62 lb)
Operating temperature	0° to 40°C (32° to 104°F)
Operating humidity	90% maximum relative humidity, noncondensing
Electromagnetic Emissions	FCC Part 15 Class B VCCI Class B EN 55 022 (CISPR 22), Class B C-Tick N10947
LAN	10BASE-T or 100BASE-Tx, RJ-45
WAN	10BASE-T or 100BASE-Tx, RJ-45
Wireless	Maximum wireless signal rate complies with the IEEE 802.11 standard. See the footnote for the previous table.
Radio data rates	Auto Rate Sensing
Data encoding standards	IEEE 802.11n version 2.0 IEEE 802.11n, IEEE 802.11g, IEEE 802.11b 2.4 GHz
Maximum computers per wireless network	Limited by the amount of wireless network traffic generated by each node (typically 50–70 nodes).
Operating frequency range	2.412–2.462 GHz (US) 2.412–2.472 GHz (Japan) 2.412–2.472 GHz (Europe ETSI)
802.11 security	WEP, WPA-PSK, WPA2-PSK, WPA-PSK + WPA2-PSK mixed mode, WPA/WPA2 Enterprise

Supplemental Information

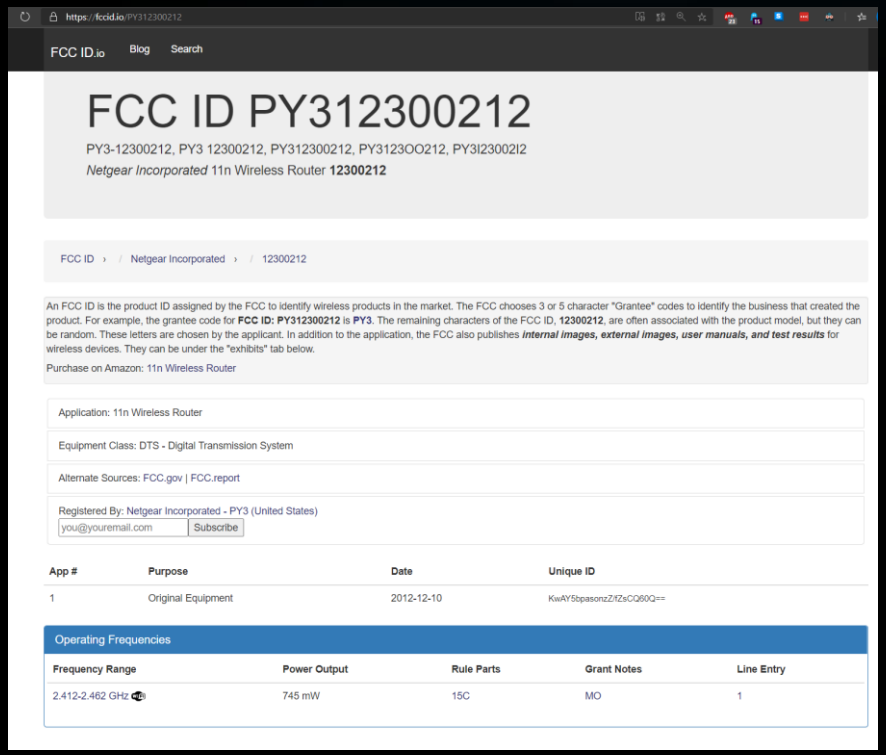
137

- Vendor's documentation
- Google
- Previous research already available
- Similar products

FCCID Lookup



https://fccid.io/PY312300212



https://fccid.io/PY312300212

FCC ID.io Blog Search

FCC ID PY312300212

PY3-12300212, PY3 12300212, PY312300212, PY3123OO212, PY3I2300212

Netgear Incorporated 11n Wireless Router **12300212**

FCC ID › Netgear Incorporated › 12300212

An FCC ID is the product ID assigned by the FCC to identify wireless products in the market. The FCC chooses 3 or 5 character "Grantee" codes to identify the business that created the product. For example, the grantee code for **FCC ID: PY312300212** is **PY3**. The remaining characters of the FCC ID, **12300212**, are often associated with the product model, but they can be random. These letters are chosen by the applicant. In addition to the application, the FCC also publishes ***internal images, external images, user manuals, and test results*** for wireless devices. They can be under the "exhibits" tab below.

Purchase on Amazon: 11n Wireless Router

Application: 11n Wireless Router

Equipment Class: DTS - Digital Transmission System

Alternate Sources: FCC.gov | FCC.report

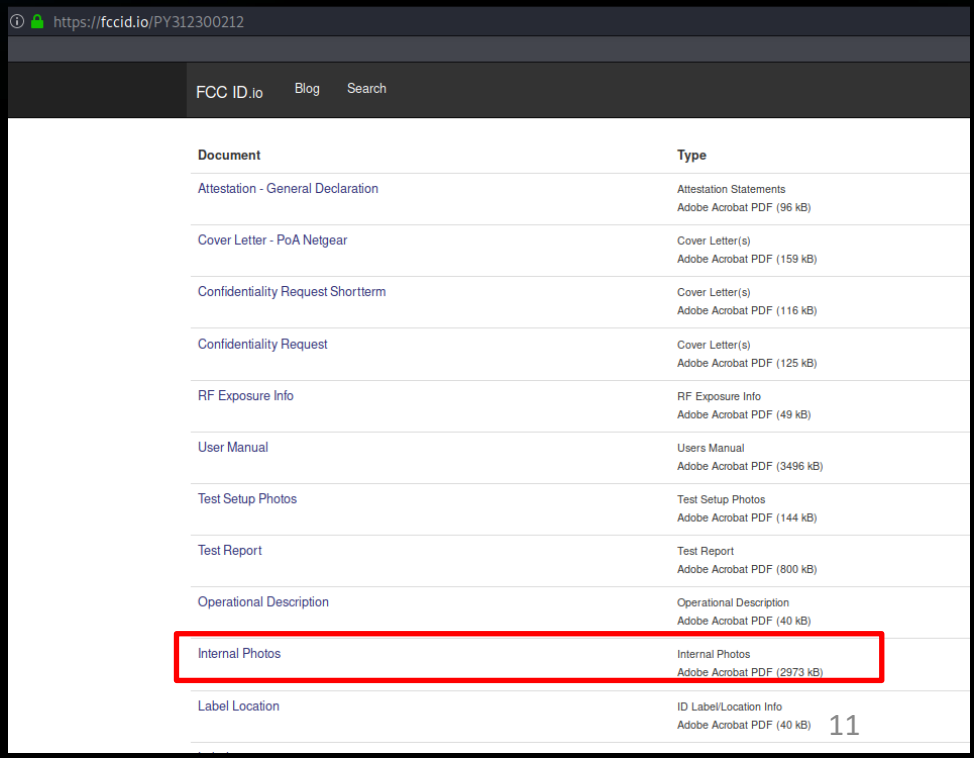
Registered By: Netgear Incorporated - PY3 (United States)

you@youremail.com Subscribe

App #	Purpose	Date	Unique ID
1	Original Equipment	2012-12-10	KwAY5bpasonzZ/fZsCQ60Q==

Operating Frequencies

Frequency Range	Power Output	Rule Parts	Grant Notes	Line Entry
2.412-2.462 GHz	745 mW	15C	MO	1



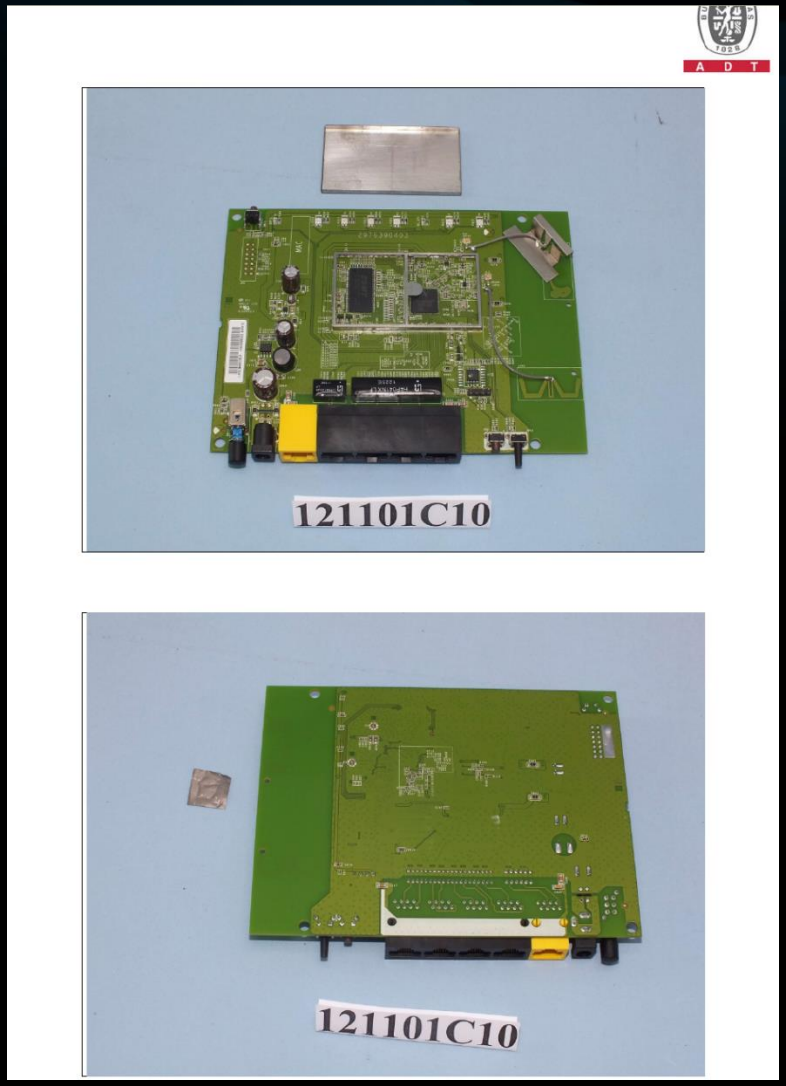
https://fccid.io/PY312300212

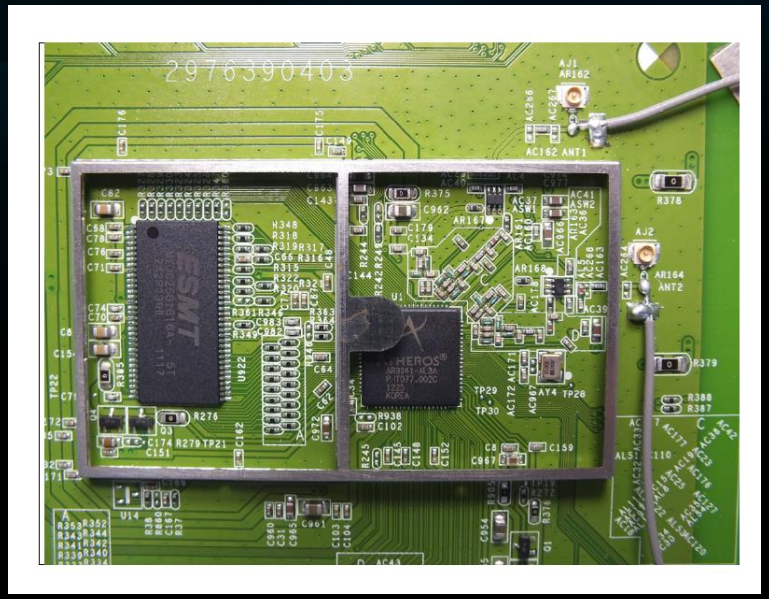
FCC ID.io Blog Search

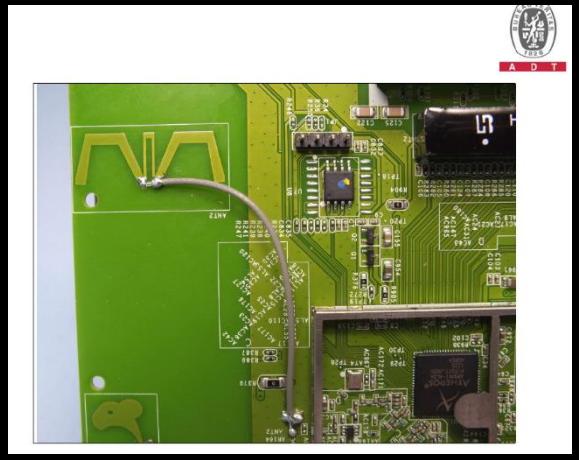
Document	Type
Attestation - General Declaration	Attestation Statements Adobe Acrobat PDF (96 kB)
Cover Letter - PoA Netgear	Cover Letter(s) Adobe Acrobat PDF (159 kB)
Confidentiality Request Shortterm	Cover Letter(s) Adobe Acrobat PDF (116 kB)
Confidentiality Request	Cover Letter(s) Adobe Acrobat PDF (125 kB)
RF Exposure Info	RF Exposure Info Adobe Acrobat PDF (49 kB)
User Manual	Users Manual Adobe Acrobat PDF (3496 kB)
Test Setup Photos	Test Setup Photos Adobe Acrobat PDF (144 kB)
Test Report	Test Report Adobe Acrobat PDF (800 kB)
Operational Description	Operational Description Adobe Acrobat PDF (40 kB)
Internal Photos	Internal Photos Adobe Acrobat PDF (2973 kB)
Label Location	ID Label/Location Info Adobe Acrobat PDF (40 kB)

FCCID Lookup => Internal Photos

https://fccid.io/PY312300212/Internal-Photos/Internal-Photos-1855783

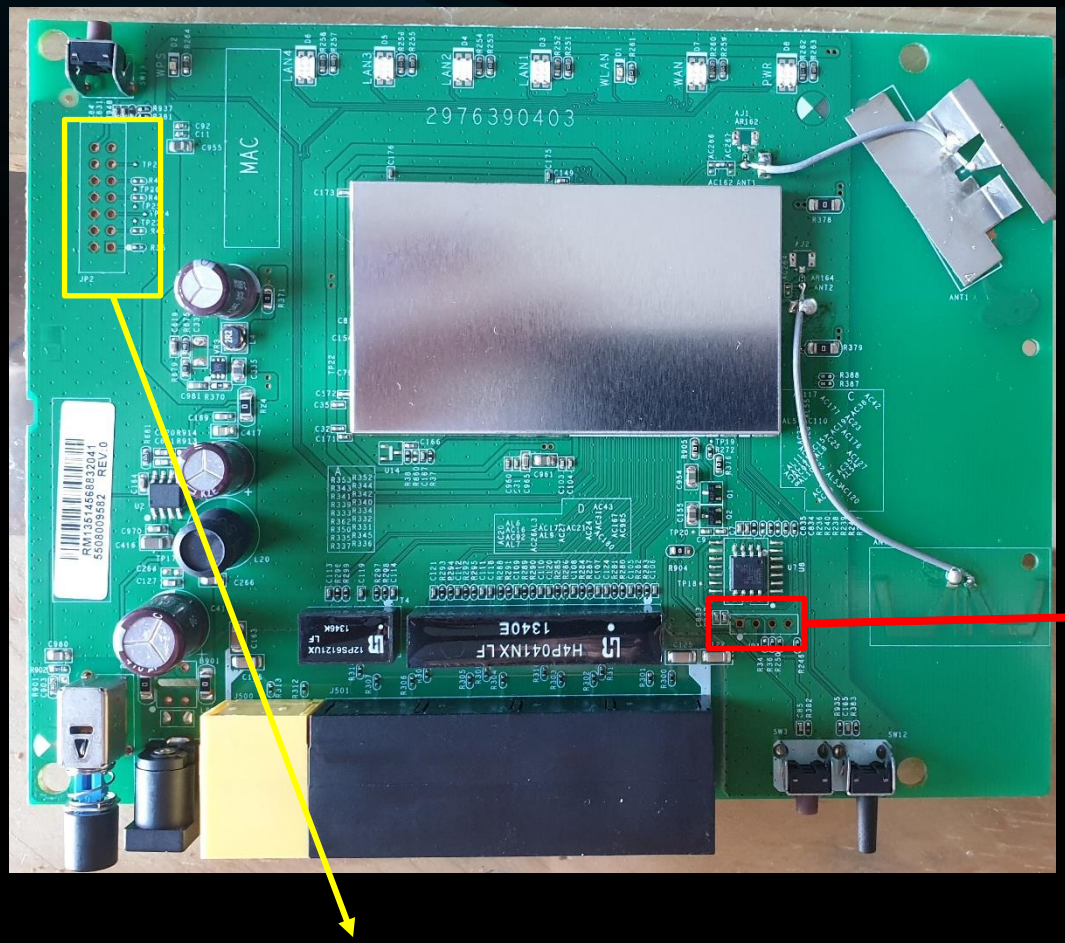






2> INTERNAL INSPECTION

Open the Device



- No need to remove metallic EMC shield for now (we have full internal photos)
- Here, no trivial indicator of debug interface is written on PCB (eg. TX, RX, TDO, TDI, TCK…)

Components Identification (1/2)

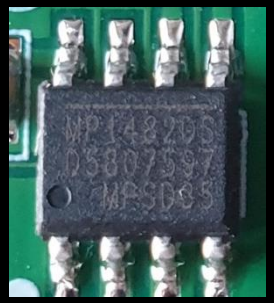
- Search references/codes on chips/components on:
  - https://www.alldatasheet.com/
  - https://www.datasheets360.com/
  - Google (filetype:pdf)

MXIC MX 25L3208E

32 M-Bit (4MB) CMOS Serial Flash

https://www.alldatasheet.com/datasheet-pdf/pdf/575458/MCNIX/MX25L3208EM2I12G.html

=> NOR Flash – Non-volatile memory



MP1482DS

2A, 18V Synchronous Rectified Step-Down Converter

https://www.alldatasheet.com/datasheet-pdf/pdf/551573/MPS/MP1482DS.html

=> Related to Power Supply, not interesting for us

Components Identification (2/2)

ESMT M13S2561616A – 5T

4M x 16 Bit x 4 Banks Double Data Rate SDRAM (32MB)

https://www.alldatasheet.com/datasheet-pdf/pdf/204934/ESMT/M13S2561616A.html

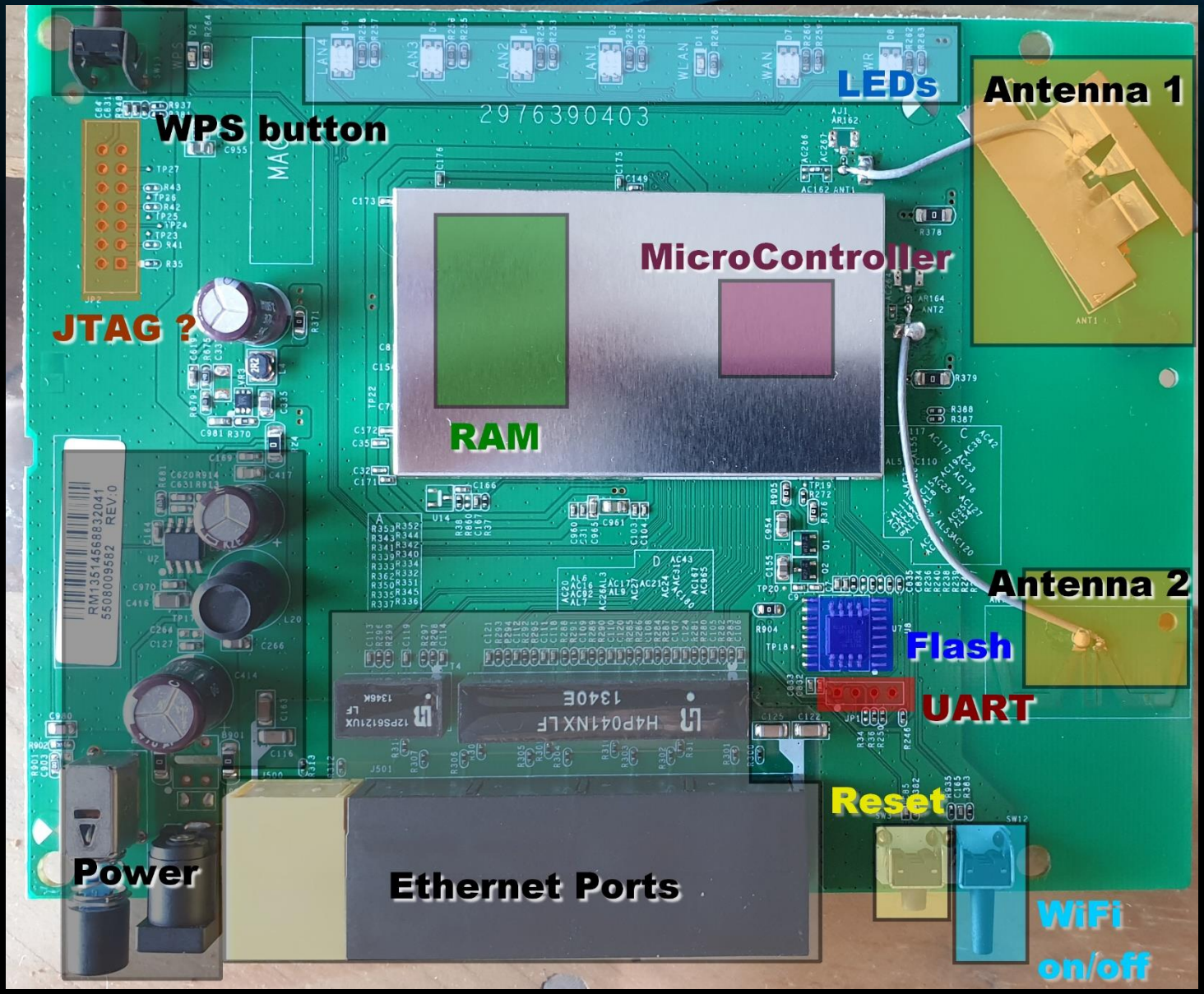
ATHEROS AR9341-AL3A

Highly-Integrated and Feature-
Rich IEEE 802.11n 2x2 2.4 GHz Premium SoC for Advanced WLAN Platforms

https://www.alldatasheet.com/datasheet-pdf/pdf/1168533/ETC1/AR9341.html

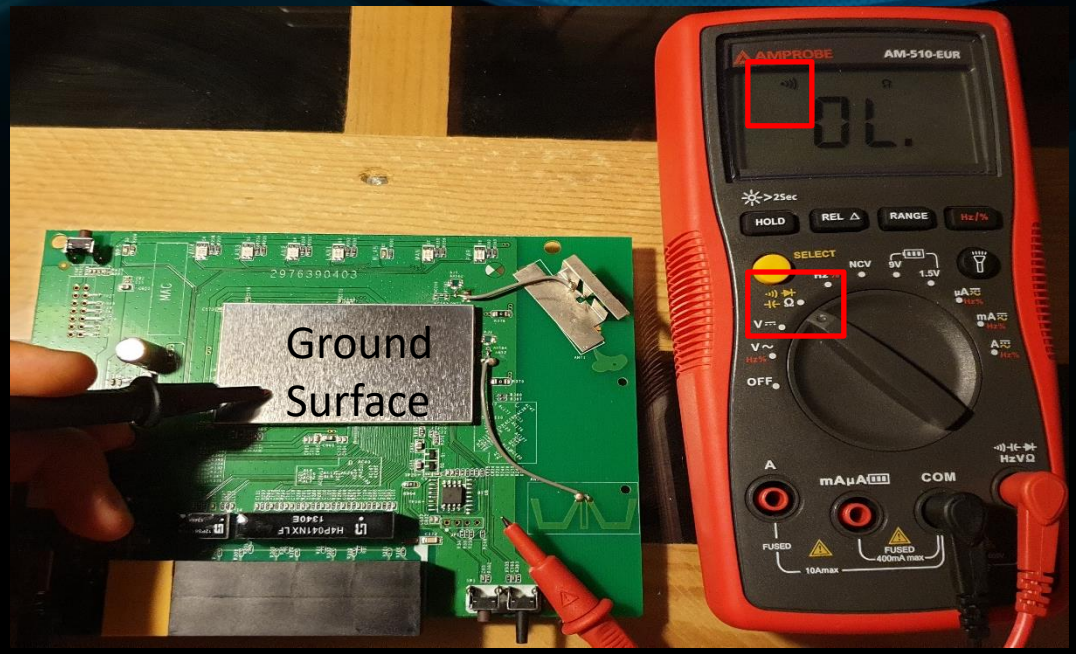
=> Micro-Controller (MCU)

PCB Overview



3> UART

UART Pins Identification - Methodology

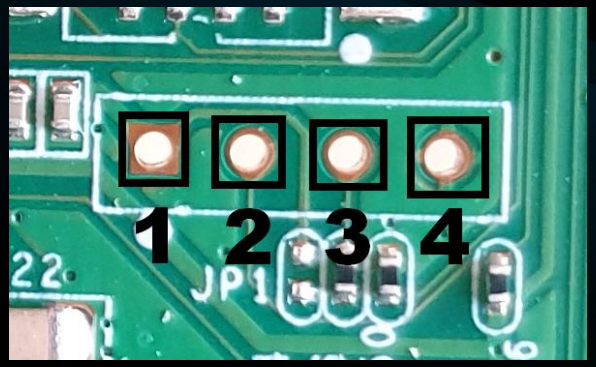


1. Multimeter in “continuity test” mode & Device powered OFF
   - Black probe on known Ground (metallic surface)
   - Red probe on each pin/pad

   ⇒ Beep indicates GND

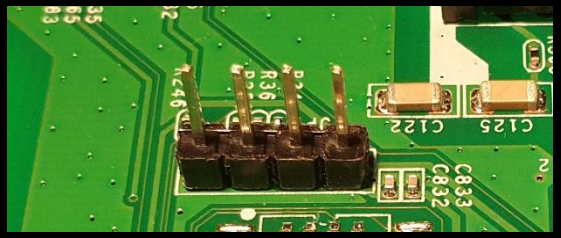
2. Multimeter in Voltmeter (DC) mode & Device powered ON
   - Black probe on known Ground
   - Red probe on each pin/pad
   - V=3.3V or 5V (constant) => Vcc
   - V=High voltage with fluctuation at boot => Probably Tx
   - V=Low voltage => Probably Rx
   - V=0V => GND (already found)

UART Pins Identification - Results

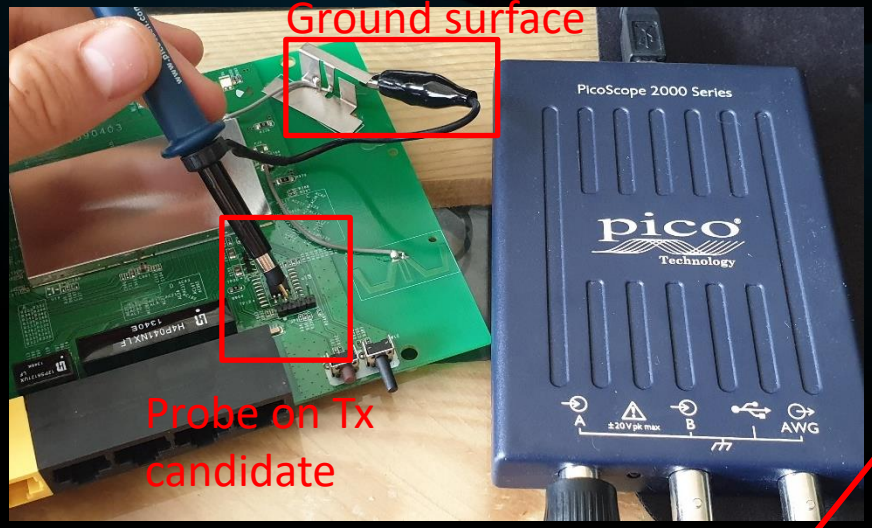


PIN	R_GND (dev OFF)	V (device ON)	Notes
1	∞	3,3V	Vcc
2	~80kΩ	1,7-2,5V (fluctuations)	Tx
3	~12kΩ	0-0,004V	Rx
4	0Ω (beep)	0V	GND

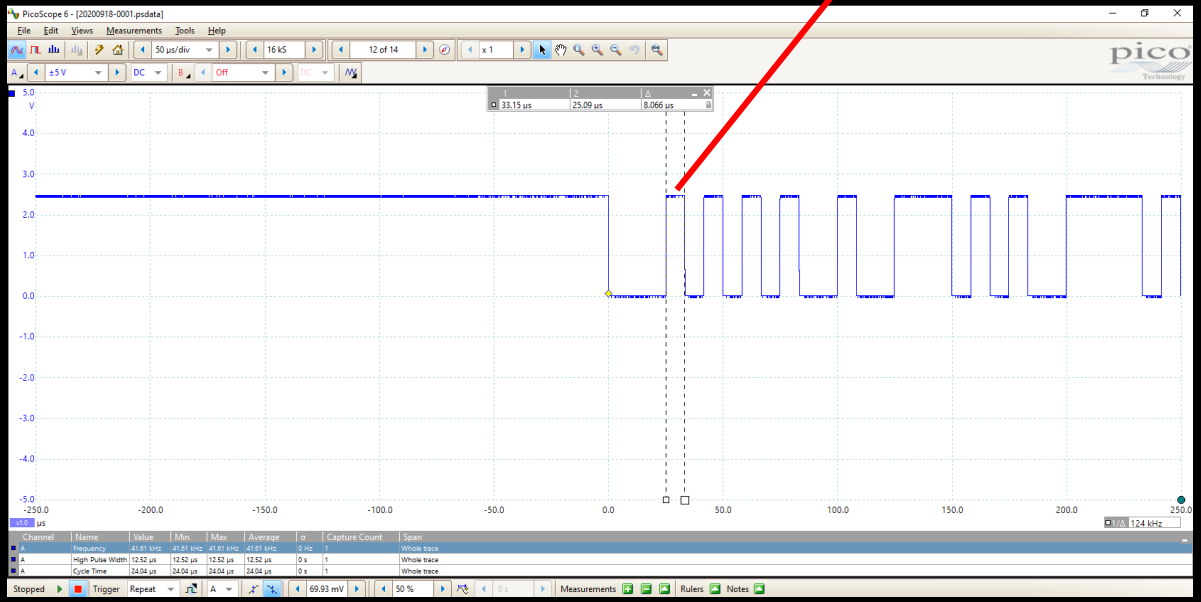
- UART is used for asynchronous communication (i.e. without a clock) => Baud rate (nb of bits / second) is required
- Common Baud rates: 9600, 38400, 19200, 57600, 115200
- We need to solder Pin headers to be able to connect using jump wires
  - Warning: bad soldering may result in bad contact or no contact at all !!



Baud rate identification with PicoScope



- Overkill here, but PicoScope can be used to find tricky points on PCB where there is data emission (eg. Isolated test points aka “TP”)
- Δ time for 1-bit ≈ 8,066 µs
  => Baud rate ≈ 1/(8,066 * 10^-6)
  ≈ 123 977



=> Closest common rate is:

115 200

Baud rate identification with Logic Analyzer

Δ time for 1-bit ≈ 8,333 µs

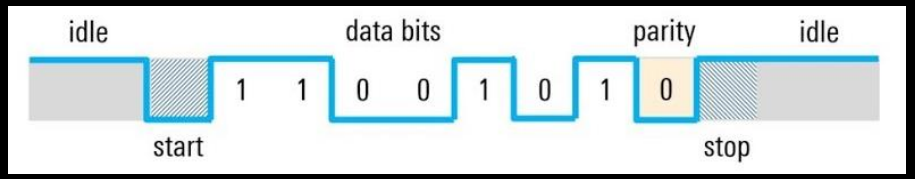
=> Baud rate ≈ 1/(8,333 * 10^-6)

≈ 120 048

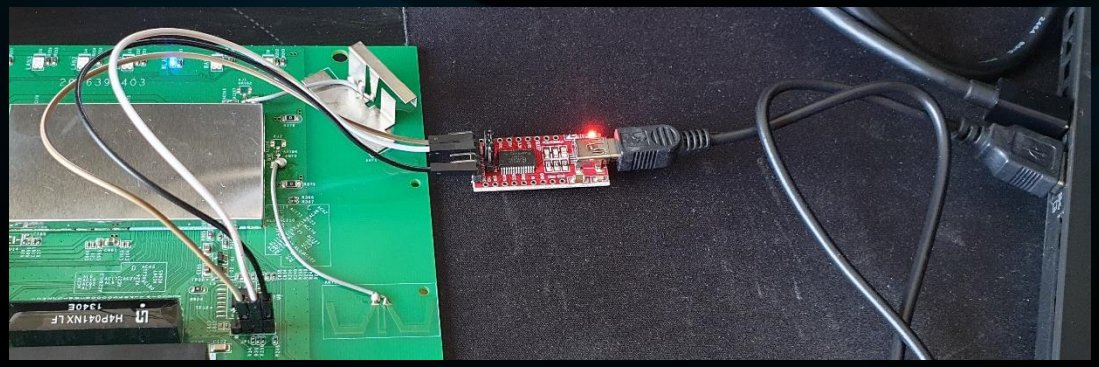
=> Closest common rate is: 115 200

Logic Analyzer can then decode data:

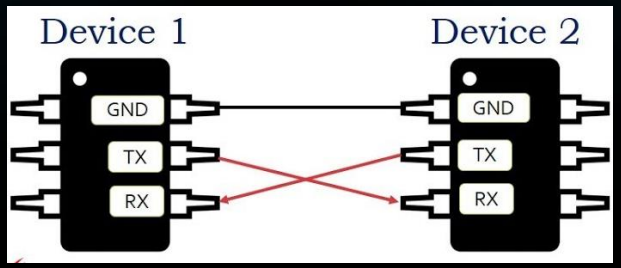
UART frame format:



Baud rate identification with Bruteforce



1. Connect Serial adapter to UART



2. Run script https://github.com/devttys0/baudrate/blob/master/baudrate.py

Will loop around common baud rates until it receives readable data

```
root@hackbox:/home/jbr/pentest-tools/hardware/baudrate# python2 baudrate.py -a

Starting baudrate detection on /dev/ttyUSB0, turn on your serial device now.
Press Ctl+C to quit.



@@@@@@@@@@@@@@@@@@@@@@ Baudrate: 115200 @@@@@@@@@@@@@@@@@@@@@@



U-Boot 1.1.4 (Nov 26

Detected baudrate: 115200

Save minicom configuration as: ^C
```

Connection to UART

- `screen /dev/ttyUSB0 115200`

```
Setting PHY...
ADDRCONF(NETDEV_UP): eth0: link is not ready
athr_gmac_ring_alloc Allocated 640 at 0x81e3dc00
athr_gmac_ring_alloc Allocated 2048 at 0x81ea6000
WASP ——→ S27 PHY MDIO
Setting Drop CRC Errors, Pause Frames and Length Error frames
ATHRS27: resetting s27
ATHRS27: s27 reset done
Setting PHY...
ADDRCONF(NETDEV_UP): eth1: link is not ready
athr_gmac_ring_free Freeing at 0x81e3d800
athr_gmac_ring_free Freeing at 0x81e55800
athr_gmac_ring_free Freeing at 0x81e3dc00
athr_gmac_ring_free Freeing at 0x81ea6000
Write Reg: 0x0000002c: Oldval = 0xfe7f007f Newval = 0xfe7f7f7f
Write Reg: 0x0000003c: Oldval = 0xcf00004e Newval = 0xce48004e
Write Reg: 0x00000104: Oldval = 0x00004804 Newval = 0x00304804
Write Reg: 0x00000204: Oldval = 0x00004004 Newval = 0x00304004
Write Reg: 0x00000304: Oldval = 0x00004004 Newval = 0x00304004
Write Reg: 0x00000404: Oldval = 0x00004004 Newval = 0x00304004
Write Reg: 0x00000504: Oldval = 0x00004004 Newval = 0x00304004
Write Reg: 0x0000003c: Oldval = 0xce48004e Newval = 0xce68004e
Write Reg: 0x00058804: Oldval = 0x00000017 Newval = 0x00000000
Write Reg: 0x00058808: Oldval = 0x00000010 Newval = 0x00000000
Write Reg: 0x0005880c: Oldval = 0x00000003 Newval = 0x00000000
Write Reg: 0x00058810: Oldval = 0x0000001c Newval = 0x00000000
Write Reg: 0x00058800: Oldval = 0x00000000 Newval = 0x00000001
Write Reg: 0x00058818: Oldval = 0x00000002 Newval = 0x00000006
Write Reg: 0x00058400: Oldval = 0x7b000521 Newval = 0x5e7ffffa
Write Reg: 0x00058404: Oldval = 0xb1f21502 Newval = 0x00000100
Write Reg: 0x00058408: Oldval = 0x88061000 Newval = 0x00000000
Write Reg: 0x0005840c: Oldval = 0x2804eb95 Newval = 0x00000001
Write Reg: 0x00058410: Oldval = 0x00000081 Newval = 0x00000001
Write Reg: 0x00058c00: Oldval = 0xefb7757e Newval = 0xffffffff
Write Reg: 0x00058c04: Oldval = 0x80c493f2 Newval = 0x0000ffff
Write Reg: 0x00058c08: Oldval = 0xf8f517e0 Newval = 0x00000000
Write Reg: 0x00058c0c: Oldval = 0xbff7dbd6 Newval = 0x00000fff
Write Reg: 0x00058814: Oldval = 0x0000004f Newval = 0x0000001f
Write Reg: 0x0005881c: Oldval = athr_gmac_ring_alloc Allocated 640 at 0x81eae000
0x00000000 Newvaathr_gmac_ring_alloc Allocated 2048 at 0x81eea000
l = 0x00000001
Write Reg: 0x000WASP ——→ S27 PHY MDIO
58000: Oldval = Setting Drop CRC Errors, Pause Frames and Length Error frames
0x00000000 NewvaSetting PHY...
l = 0x00000000
Write Reg: 0x00058004: Oldval = 0x08049100 Newval = 0x11f00000
Write Reg: 0x00058008: Oldval = 0x00000000 Newval = 0x00000000
init.enet: Default WAN MAC is : C4:04:15:99:6A:CB
```

```
Copyright 2005 by Johnny Egeland <johnny@rlo.org>
Modified by Tos Xu for IGMP snooping in April, 2009.
Distributed under the GNU GENERAL PUBLIC LICENSE, Version 2 - check GPL.txt

MTD partition not found.
Boot up procedure is Finished!!!

Please press Enter to activate this console. Sending discover...
Sending discover...
Sending discover...


BusyBox v1.4.2 (2013-11-12 17:41:20 CST) Built-in shell (ash)
Enter 'help' for a list of built-in commands.

 _______ ________ __         __
|       |.-----.|  |_.-----.|  |--.
|   -   ||  _  ||   _|  -__||     |
|_______||   __||____|_____||__|__|
         |__| W I R E L E S S   F R E E D O M
 KAMIKAZE (bleeding edge, r18571) ----------
  * 10 oz Vodka       Shake well with ice and strain
  * 10 oz Triple sec  mixture into 10 shot glasses.
  * 10 oz lime juice  Salute!
 ---------------------------------------------------

root@WNR2000v4:/# Sending discover...
Sending discover...
Sending discover...

root@WNR2000v4:/# Sending discover...
Sending discover...

root@WNR2000v4:/#
root@WNR2000v4:/# Sending discover...
ls
bin                     jffs                    sbin
default_language_version lib                    sys
dev                     mnt                     tmp
etc                     module_name             usr
firmware_region         proc                    var
firmware_version        rom                     www
hardware_version        root
root@WNR2000v4:/# cat firmware_version
V1.0.0.50
root@WNR2000v4:/# Sending discover...
Sending discover...
Sending discover...
```

- Well-known open-source OS for embedded devices (based on Linux) OpenWrt is used
- UART connection gives a root shell => full access to filesystem

4> BOOTLOADER (via UART)

Info Gathering using Bootloader (1/2)

- Bootloader in use: U-Boot (very popular for embedded devices)
- Boot logs analysis (record all data sent by Tx & look for interesting stuff)
- Enter in Bootloader menu (press key at boot) and look for available commands

```
U-Boot 1.1.4 (Nov 26 2012 - 15:58:42)

DNT HW ID: 29763904 flash 4MB RAM 32MB U-boot dni29 V0.5
DRAM:
sri
Wasp 1.3
wasp_ddr_initial_config(281): Wasp (16bit) ddr1 init
Tap value selected = 0×f [0×0 - 0×1f]
32 MB
Top of RAM usable for U-Boot at: 82000000
Reserving 218k for U-Boot at: 81fc8000
Reserving 192k for malloc() at: 81f98000
Reserving 44 Bytes for Board Info at: 81f97fd4
Reserving 36 Bytes for Global Data at: 81f97fb0
Reserving 128k for boot params() at: 81f77fb0
Stack Pointer at: 81f77f98
Now running in RAM - U-Boot at: 81fc8000
Flash Manuf Id 0×c2, DeviceId0 0×20, DeviceId1 0×16
flash size 4MB, sector count = 64
Flash: 4 MB
*** Warning - bad CRC, using default environment

In:    serial
Out:   serial
Err:   serial
Net:   ag934x_enet_initialize...
Fetching MAC Address from 0×81fecd38
Fetching MAC Address from 0×81fecd38
 wasp  reset mask:c03300
WASP ——> S27 PHY
: cfg1 0×80000000 cfg2 0×7114
eth0: c4:04:15:99:6a:cb
s27 reg init
athrs27_phy_setup ATHR_PHY_CONTROL 4 :1000
athrs27_phy_setup ATHR_PHY_SPEC_STAUS 4 :10
eth0 up
WASP ——> S27 PHY
: cfg1 0×f cfg2 0×7214
eth1: c4:04:15:99:6a:ca
s27 reg init lan
ATHRS27: resetting s27
ATHRS27: s27 reset done
athrs27_phy_setup ATHR_PHY_CONTROL 0 :1000
athrs27_phy_setup ATHR_PHY_SPEC_STAUS 0 :10
athrs27_phy_setup ATHR_PHY_CONTROL 1 :1000
athrs27_phy_setup ATHR_PHY_SPEC_STAUS 1 :10
athrs27_phy_setup ATHR_PHY_CONTROL 2 :1000
athrs27_phy_setup ATHR_PHY_SPEC_STAUS 2 :10
athrs27_phy_setup ATHR_PHY_CONTROL 3 :1000
athrs27_phy_setup ATHR_PHY_SPEC_STAUS 3 :10
eth1 up
eth0, eth1
Hit any key to stop autoboot:
ar7240>
```

```
bootm   - boot application image from memory
bootp   - boot image via network using BootP/TFTP protocol
cmp     - memory compare
coninfo - print console devices and information
cp      - memory copy
crc32   - checksum calculation
dhcp    - invoke DHCP client to obtain IP/boot params
echo    - echo args to console
erase   - erase FLASH memory
ethreg    - S26 PHY Reg rd/wr  utility
exit    - exit script
flinfo  - print FLASH memory information
fw_recovery - start tftp server to recovery dni firmware image.
go      - start application at address 'addr'
help    - print online help
iminfo  - print header information for application image
imls    - list all images found in flash
itest   - return true/false on integer compare
loop    - infinite loop on address range
macset - Set ethernet MAC address
macshow - Show ethernet MAC addresses
md      - memory display
mii     - MII utility commands
mm      - memory modify (auto-incrementing)
mtest   - simple RAM test
mw      - memory write (fill)
nm      - memory modify (constant address)
nmrp - start nmrp mechanism to upgrade firmware-image or string-table.
nor_fw_integrity_check - verify firmware checksum in NOR
nor_two_part_fw_integrity_check - verify firmware checksum in NOR
pci     - list and access PCI Configuration Space
ping    - send ICMP ECHO_REQUEST to network host
pll cpu-pll dither ddr-pll dither - Set to change CPU & DDR speed
pll erase
pll get
printenv- print environment variables
```

Info Gathering using Bootloader (2/2)

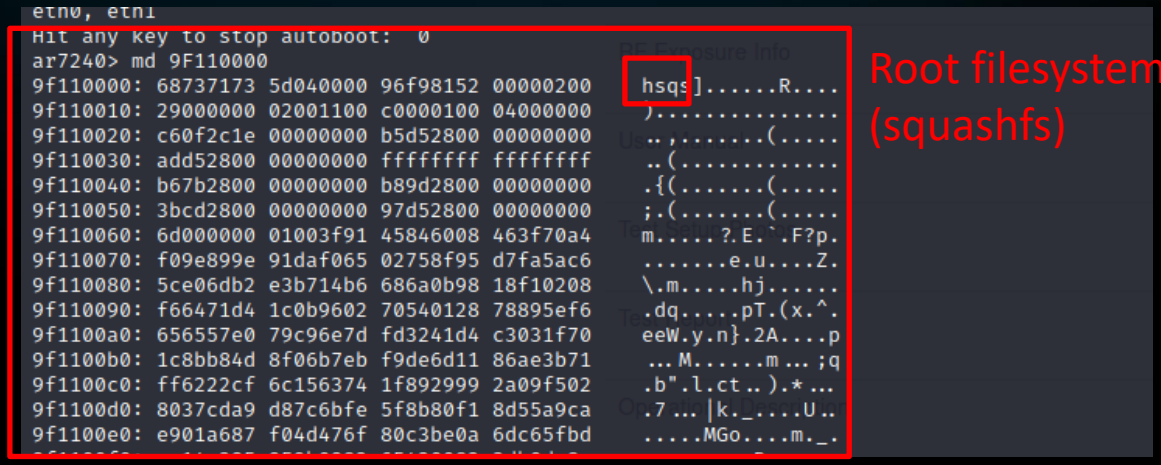
```
ar7240>
ar7240> bdinfo
boot_params = 0x81F77FB0
memstart    = 0x80000000
memsize     = 0x02000000
flashstart  = 0x9F000000
flashsize   = 0x00400000
flashoffset = 0x0002E310
ethaddr     = 00:AA:BB:CC:DD:EE
ip_addr     = 192.168.1.1
baudrate    = 115200 bps
ar7240>
ar7240> board_ssid_show
board_ssid : NETGEAR32
ar7240>
ar7240> coninfo
List of available devices:
serial   80000003 SIO stdin stdout stderr
ar7240>
ar7240> imls
Image at 9F040000:
   Image Name:   MIPS OpenWrt Linux-2.6.31
   Created:      2013-11-12   9:49:12 UTC
   Image Type:   MIPS Linux Kernel Image (lzma compressed)
   Data Size:    804410 Bytes = 785.6 kB
   Load Address: 80002000
   Entry Point:  801e68d0
   Verifying Checksum ... OK
ar7240>
ar7240> version

U-Boot 1.1.4 (Nov 26 2012 - 15:58:42)
ar7240>
ar7240> printenv
bootargs=console=ttyS0,115200 root=31:02 rootfstype=jffs2 init=/sbin/init mtdparts
bootcmd=sleep 1; nmrp; nor_two_part_fw_integrity_check 0x9f040000; bootm 0x9f0400
bootdelay=1
baudrate=115200
ethaddr=0x00:0xaa:0xbb:0xcc:0xdd:0xee
ipaddr=192.168.1.1
serverip=192.168.1.10
dir=
lu=tftp 0x80060000 ${dir}u-boot.bin&&erase 0x9f000000 +$filesize;cp.b $fileaddr 0
lf=tftp 0x80060000 ${dir}db12x${bc}-jffs2&&erase 0x9f050000 +0x630000;cp.b $filead
lk=tftp 0x80060000 ${dir}vmlinux${bc}.lzma.uImage&&erase 0x9f680000 +$filesize;cp.
stdin=serial
stdout=serial
stderr=serial
ethact=eth0

Environment size: 739/65532 bytes
ar7240>
```

- Analysis Results Notes:

```
Bootloader:          U-Boot 1.1.4 (Nov 26 2012 - 15:58:42)

OS:                  MIPS OpenWrt Linux-2.6.31
Arch:                MIPS
GCC version:         gcc version 4.3.3 (GCC)

U-Boot addr:         81fc8000
Firmware addr:       9F040000

RAM device:
RAM start addr:      80000000
RAM size:            02000000 (32MB)

Flash device:        29763904 flash 4MB
Flash start addr:    9F000000
Flash size:          00400000 (4MB)

Image Size:          804410 Bytes = 785.6 kB
Load Address:        80002000 (in RAM)
Entry Point:         801e68d0 (in RAM)

Root filesystem:     squashfs version 4.0 (2009/01/31)
MTD partitions:
0x000000000000-0x000000030000 : "u-boot"            192kB
0x000000030000-0x000000040000 : "u-boot-env"        64kB
0x000000040000-0x000000110000 : "kernel"            832kB
0x000000110000-0x0000003a0000 : "rootfs"            2624kB
0x0000003a0000-0x000000400000 : "rootfs_data"
0x0000003a0000-0x0000003c0000 : "language"          128kB
0x0000003c0000-0x0000003d0000 : "pot"               64kB
0x0000003d0000-0x0000003e0000 : "traffic_meter"     64kB
0x0000003e0000-0x0000003f0000 : "config"            64kB
0x0000003f0000-0x000000400000 : "art"               64kB
0x000000040000-0x0000003a0000 : "firmware"          3456kB

BusyBox version:     BusyBox v1.4.2 (2013-11-12 17:41:20 CST)
```

Dump Firmware via U-Boot

- Command `md` (memory display) can be used to display memory areas (including Flash where firmware is stored)

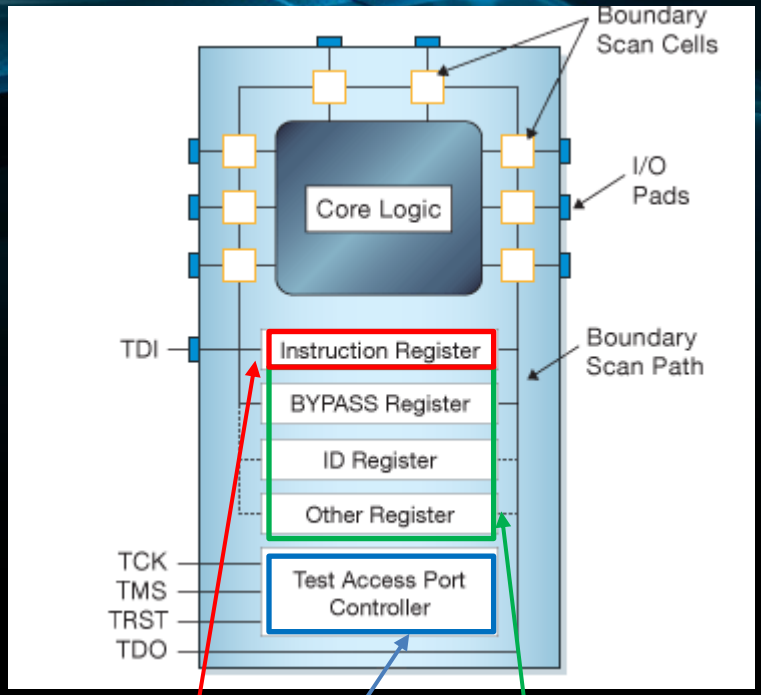


- Dump Firmware:
  1. Full Flash dump (4MB) can be done with command: `md.b 9F000000 0x400000`
  2. Record all outputs to file. Wait…
  3. Convert full hexdump to binary (https://github.com/gmbnomis/uboot-mdb-dump)

5> JTAG

JTAG (IEEE 1149.1)

- One of most widely deployed debug standards for embedded devices
- Provides direct interface to hardware on PCB, such as Flash or RAM
- JTAG Pins:
  - **TDI**: Data In
  - **TDO**: Data Out
  - **TMS**: Test Mode Select (used to control state of TAP controller)
  - **TCK**: Clock (for synchronization)
  - TRST: Test Reset (optional because possible to reset using TMS)



Indicates to TAP controller which Data Register will be used

TAP controller:

- Handles state machine logic (via TMS value)
- Implements core JTAG instructions
- Handle Clock

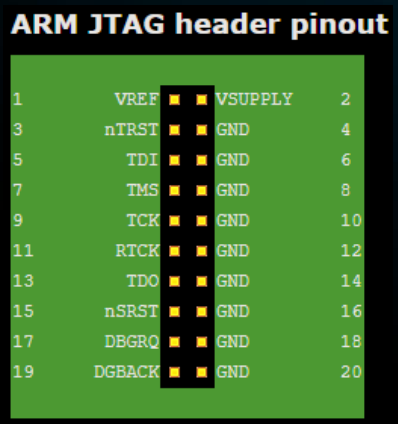
Data Registers

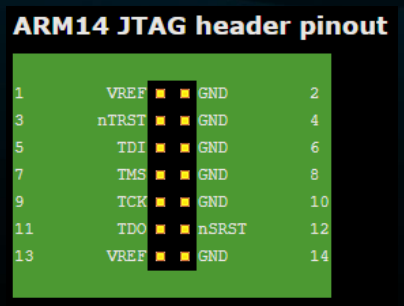
- BYPASS: 1-bit reg. used to pass data from TDI to TDO (when IR = BYPASS)
- ID: stores device ID

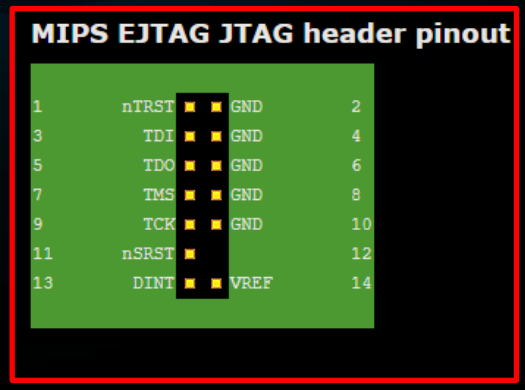
https://www.optiv.com/explore-optiv-insights/downloads/jtag-interface-attackers-perspective

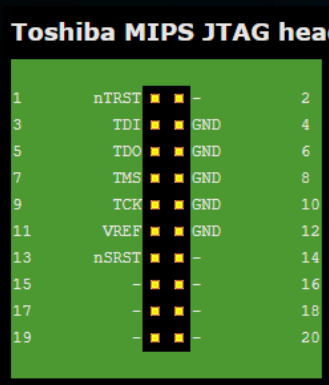
JTAG Pins Identification – Standards Pinouts

- Some standard pinouts are known: http://www.jtagtest.com/pinouts/





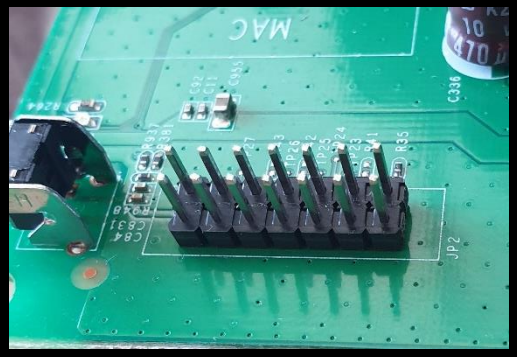




...

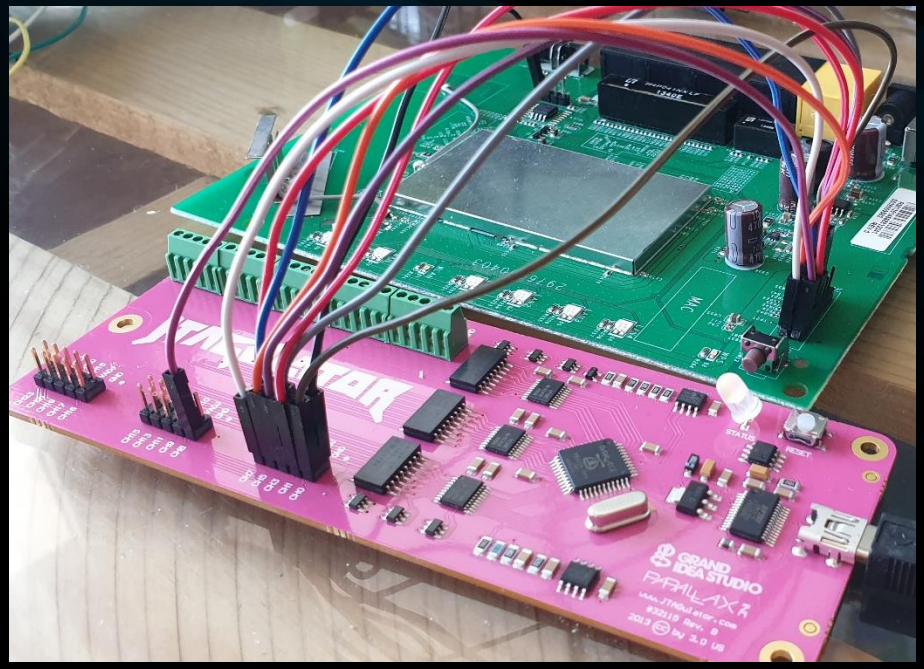
Good candidate for following reasons:

- Same number of pins
- Same position of GND (continuity test with multimeter)
- Device architecture is MIPS



- For further tests on this interface, we solder Pin headers to be able to connect with jump wires

JTAG Pins Identification – JTAGulator



1. Connect GND -> GND
2. Connect CH0..CHX -> Pins to test on PCB
3. Connect to JTAGulator (baud rate 115200)
4. Set target voltage (here 3.3V)
5. Run IDCODE scan (fast) to detect:
   - TDO
   - TCK
   - TMS
6. Run BYPASS scan (slow) to detect remaining TDI pin

Author's demo:
https://www.youtube.com/watch?v=GgMOBhmEJXA

But here: FAIL ☹

No JTAG pins found on this interface !

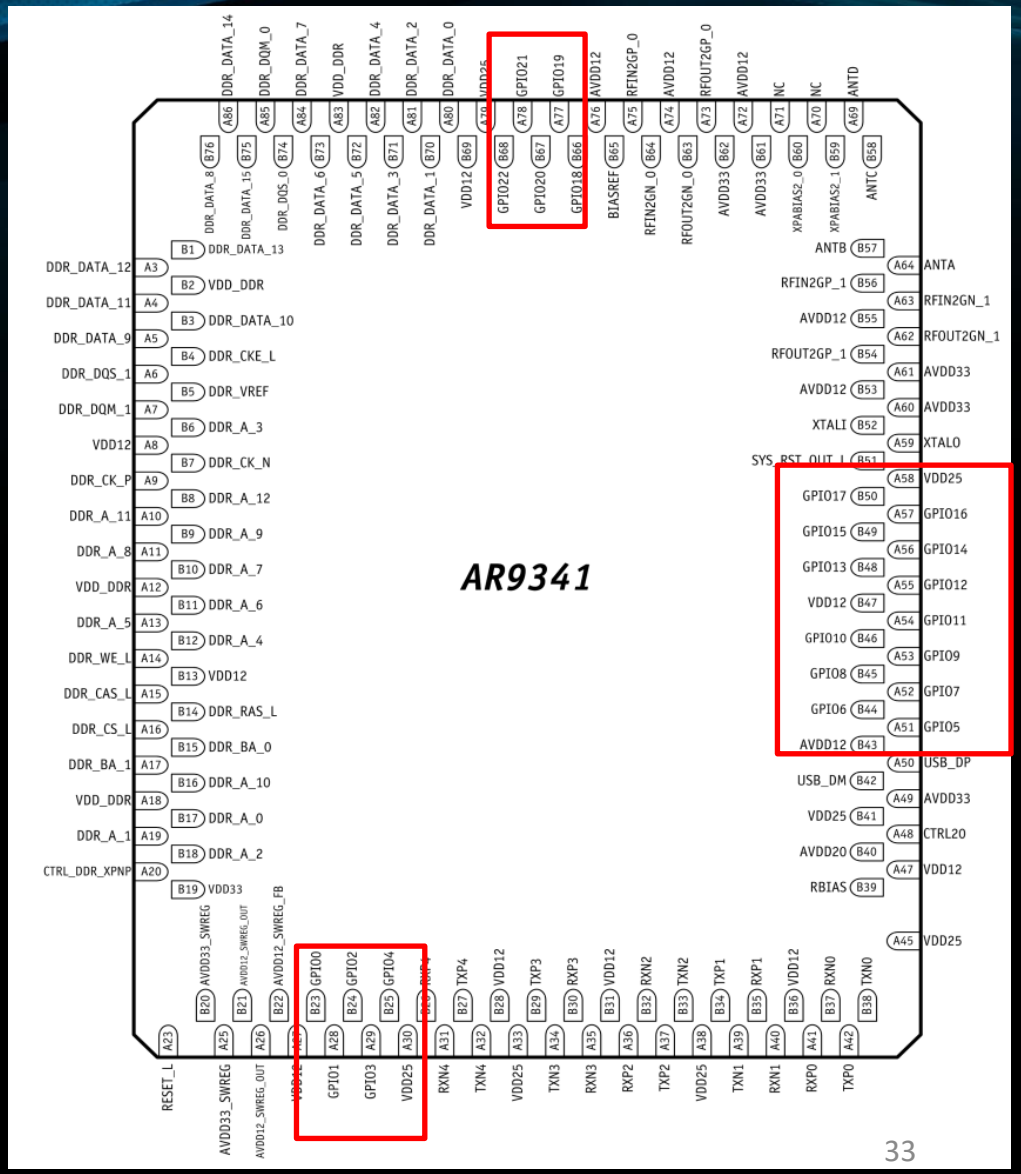
Advanced JTAG Research (1/2)

- Finding JTAG may require analyzing PCB traces:
  1. Check Micro-Controller pinouts from datasheet
  2. Identify pins that can be used for JTAG
  3. Follow traces from those pins (visual inspection + continuity test with multimeter)

- Here we see that GPIO tagged pins can be used for JTAG (but also many other stuff)

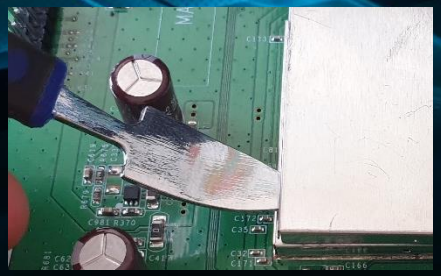
Table 1-1. **Signal to Pin Relationships and Descriptions (continued)**

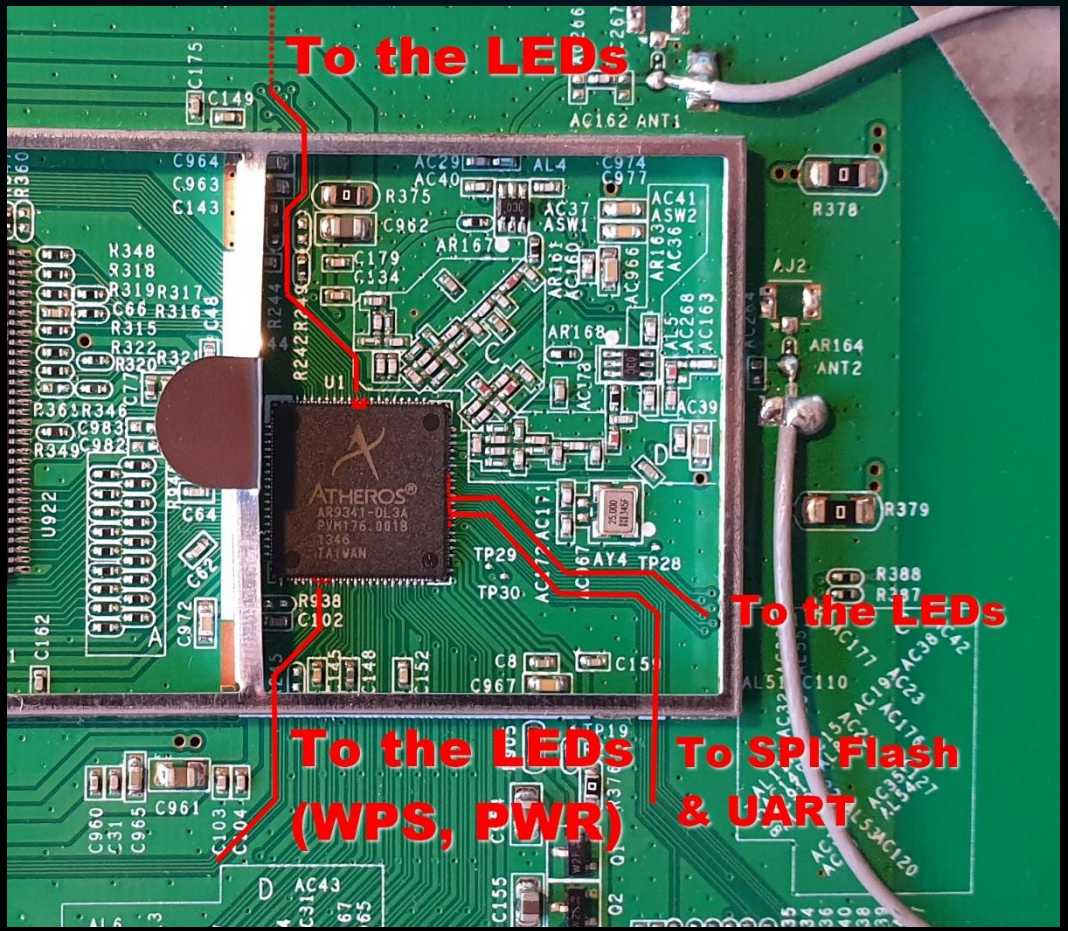
Signal Name	Pins	Type	Description
GPIO			
GPIO0	B23	I/O	General purpose I/O, programmable, can to be used as JTAG, SPI, I2S, SLIC, UART, LED control.
GPIO1	A28	I/O	
GPIO2	B24	I/O	
GPIO3	A29	I/O	
GPIO4	B25	I/O	
GPIO5	A51	I/O	
GPIO6	B44	I/O	



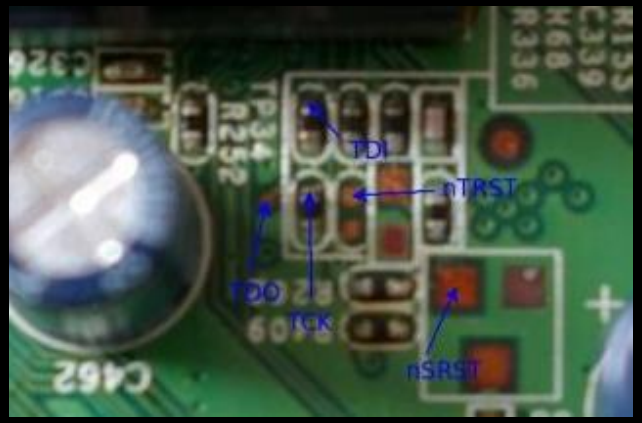
Advanced JTAG Research (2/2)

- Micro-Controller is under EMC shield
- Analysis of PCB traces from GPIO pins:





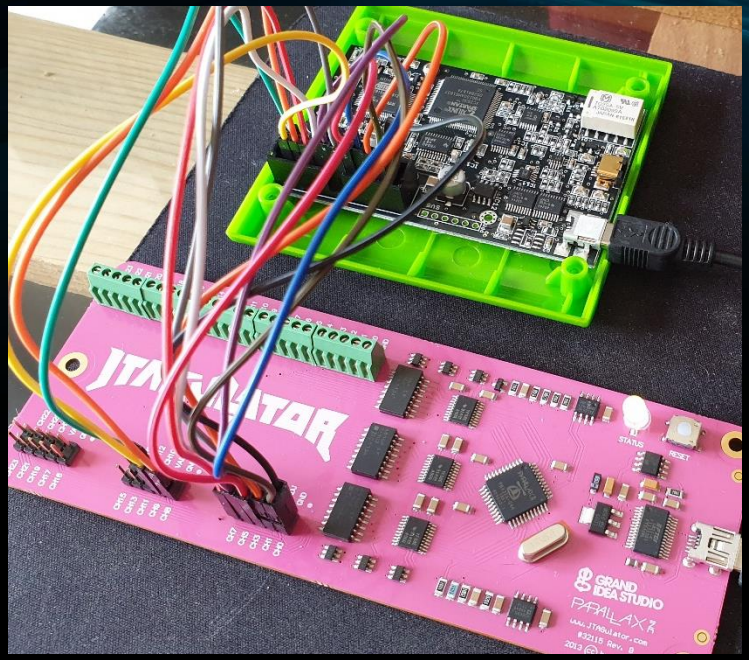
- Here I could not find JTAG on this PCB
- Maybe it has been disabled (often the case in production)
- JTAG can be **very tricky** to find => Example on previous version of router (WNR2000v1):



https://openwrt.org/toh/netgear/wnr2000

Testing on Device with known JTAG

- For experimentation purpose: Proxmark3's PCB has a known JTAG interface
- Let's find out JTAG pinout as if it was not available



```
                                    UU  LLL
 JJJ  TTTTTTT AAAAA  GGGGGGGGGG   UUUU LLL   AAAAA TTTTTTTT OOOOOOO  RRRRRRRRR
 JJJJ TTTTTTT AAAAAA GGGGGGG      UUUU LLL  AAAAAA TTTTTTTT OOOOOOO  RRRRRRRR
 JJJJ  TTTT  AAAAAAA GGG      UUU  UUUU LLL  AAA AAA   TTT  OOOO OOO  RRR RRR
 JJJJ  TTTT  AAA AAA GGG  GGG UUUU UUUU LLL AAA  AAA   TTT  OOO  OOO  RRRRRR
 JJJJ  TTTT  AAA  AA GGGGGGGGG UUUUUUUU LLLLLLLL AAAA  TTT OOOOOOOOO  RRR RRR
  JJJ  TTTT AAA   AA GGGGGGGGG UUUUUUUU LLLLLLLL AAA  TTT OOOOOOOOO  RRR RRR
  JJJ  TT                GGG             AAA                       RR RRR
 JJJ                      GG             AA                           RRR
JJJ                        G             A                             RR

        Welcome to JTAGulator. Press 'H' for available commands.

:H
JTAG Commands:
I   Identify JTAG pinout (IDCODE Scan)
B   Identify JTAG pinout (BYPASS Scan)
D   Get Device ID(s)
T   Test BYPASS (TDI to TDO)

UART Commands:
U   Identify UART pinout
P   UART passthrough

General Commands:
V   Set target I/O voltage (1.2V to 3.3V)
R   Read all channels (input)
W   Write all channels (output)
J   Display version information
H   Display available commands
:V
Current target I/O voltage: Undefined
Enter new target I/O voltage (1.2 - 3.3, 0 for off): 3.3
New target I/O voltage set: 3.3
Ensure VADJ is NOT connected to target!
:I
Enter number of channels to use (3 - 24): 11
Ensure connections are on CH10..CH0.
Possible permutations: 990
Press spacebar to begin (any other key to abort)...
JTAGulating! Press any key to abort.....
TDI: N/A
TDO: 3
TCK: 5
TMS: 6
.......
IDCODE scan complete!
:
```

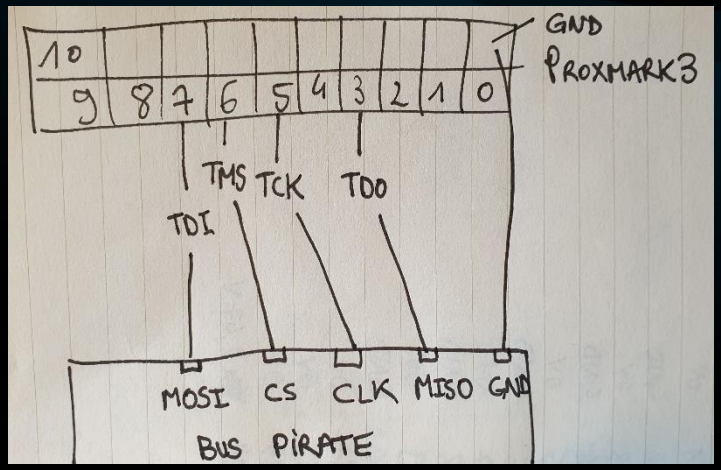
```
:B
Enter number of channels to use (4 - 24): 11
Ensure connections are on CH10..CH0.
Possible permutations: 7920
Press spacebar to begin (any other key to abort)...
JTAGulating! Press any key to abort.....................
.....................................................
.....................................................
TDI: 7
TDO: 3
TCK: 5
TMS: 6
Number of devices detected: 1
.....................................................
.....................................................
.........................
BYPASS scan complete!
:
```

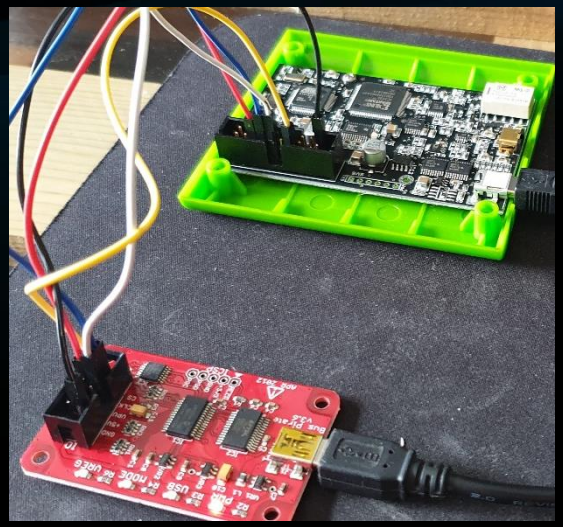
Testing

```
:T
Enter new TDI pin [0]: 7
Enter new TDO pin [0]: 3
Enter new TCK pin [0]: 5
Enter new TMS pin [0]: 6
Enter number of devices in JTAG chain [0]: 1
All other channels set to output HIGH.

Pattern in to TDI:    01100111000101011110110010011000
Pattern out from TDO: 01100111000101011110110010011000
Match!
:D
TDI not needed to retrieve Device ID.
Enter new TDO pin [3]:
Enter new TCK pin [5]:
Enter new TMS pin [6]:
Enter number of devices in JTAG chain [1]:
All other channels set to output HIGH.

Device ID: 0011 1111000011110000 11110000111 1 (0x3F0F0F0F)
→ Manufacturer ID: 0x787
→ Part Number: 0xF0F0
→ Version: 0x3

IDCODE listing complete!
:
```

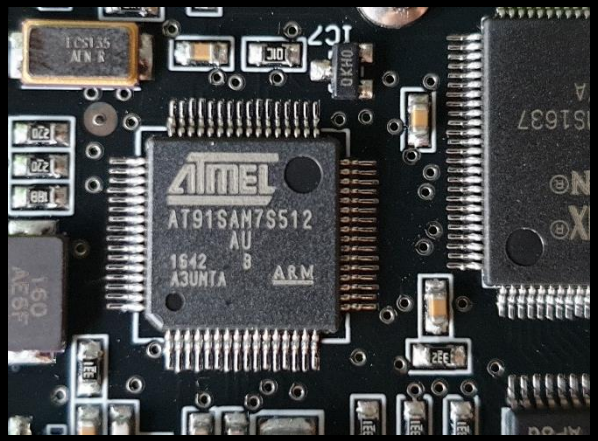
Dumping Firmware via JTAG (1/3)

1. Based on discovered JTAG pinout, connect Bus Pirate as follows:





2. Identify Micro-Controller on PCB. On Proxmark3 => AT91SAM7S512 (ARM)



3. Search for OpenOCD's configuration for this Micro-Controller

```
jbr@hackbox:/usr/share/openocd/scripts/target$ ls ./at91sam7
at91sam7a2.cfg     at91sam7se512.cfg  at91sam7sx.cfg     at91sam7x256.cfg   at91sam7x512.cfg
```

Dumping Firmware via JTAG (2/3)

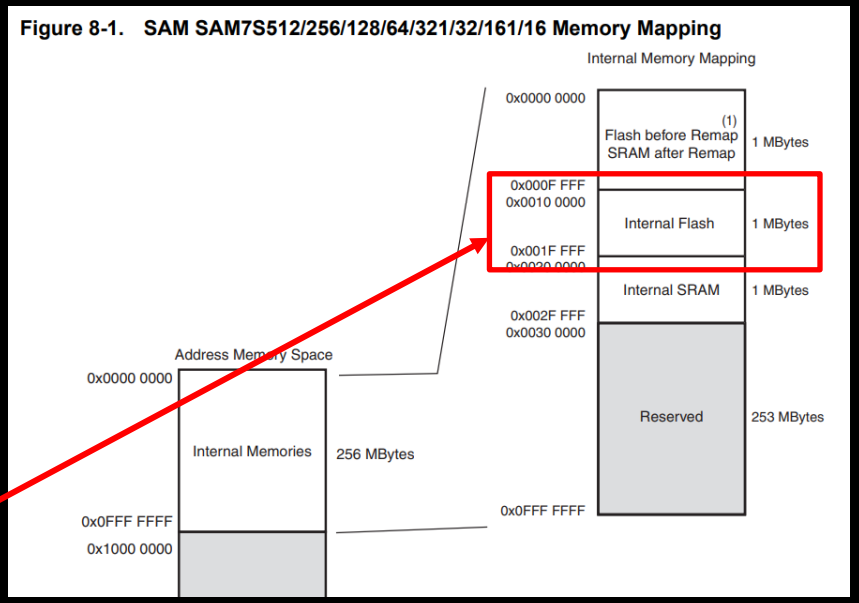
4. Run OpenOCD with config for Bus Pirate + config for Micro-Controller:

```
root@hackbox:/home/jbr/Projet-Netgear#
root@hackbox:/home/jbr/Projet-Netgear# openocd -f ./buspirate.cfg -f /usr/share/openocd/scripts/target/at91sam7x512.cfg
Open On-Chip Debugger 0.10.0+dev-snapshot (2020-08-19-07:12)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
srst_only separate srst_gates_jtag srst_open_drain connect_deassert_srst

Info : auto-selecting first available session transport "jtag". To override use 'transport select <transport>'.
Info : Listening on port 6666 for tcl connections
Info : Listening on port 4444 for telnet connections
Info : Buspirate JTAG Interface ready!
Info : This adapter doesn't support configurable speed
Info : JTAG tap: sam7x512.cpu tap/device found: 0x3f0f0f0f (mfg: 0x787 (<unknown>), part: 0xf0f0, ver: 0x3)
Info : Embedded ICE version 1
Info : sam7x512.cpu: hardware has 2 breakpoint/watchpoint units
Info : starting gdb server for sam7x512.cpu on 3333
Info : Listening on port 3333 for gdb connections
```

5. We need to know which memory region we want to dump ?
   Proxmark3 has no external Flash, and the firmware is directly stored on MCU (no OS, no filesystem like on our router)

=> Check MCU datasheet for Memory Mapping

Internal Flash mapped at:
0x100000-0x1FFFFF (size=0x100000 bytes)

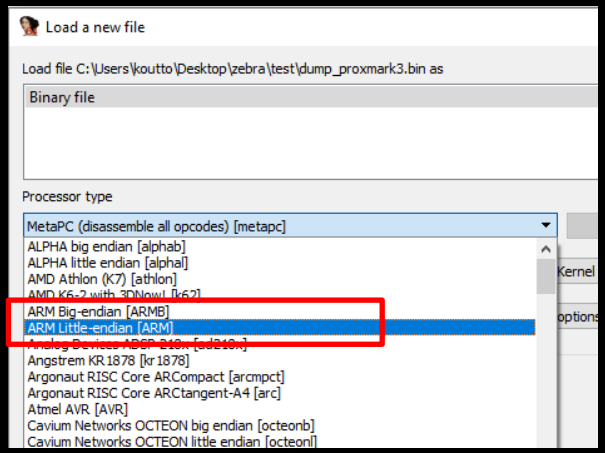


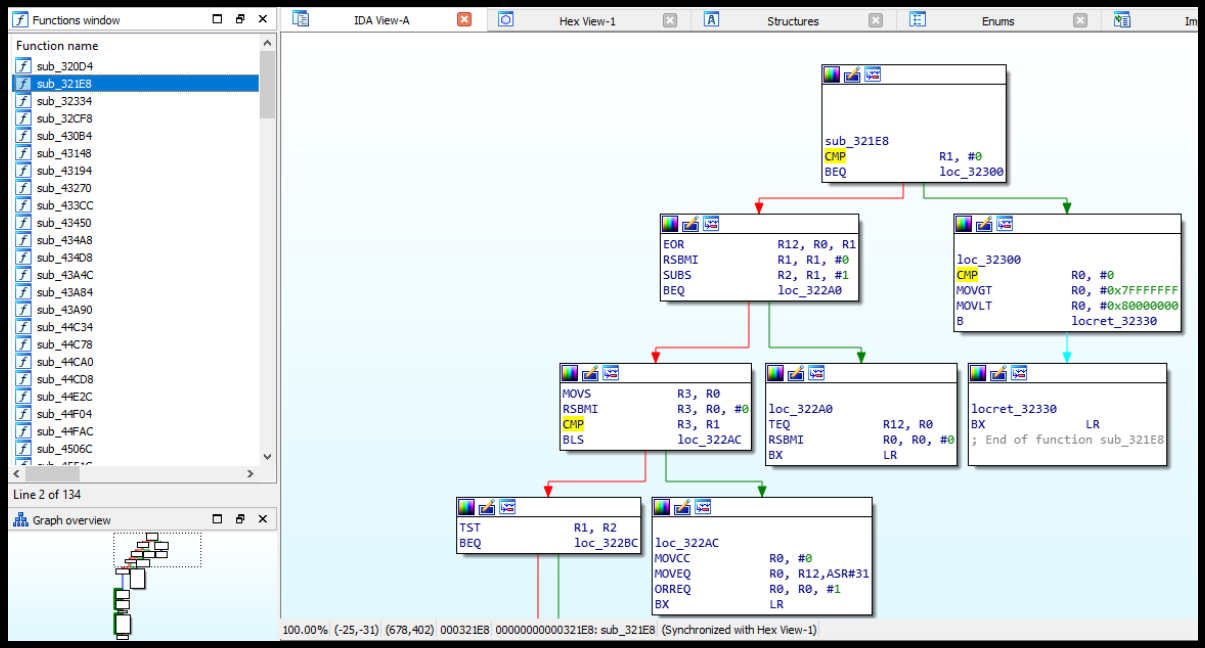
Dumping Firmware via JTAG (3/3)

6. Connect to localhost:4444 and run `dump_image` with correct offset & size:

```
jbr@hackbox:~$ telnet localhost 4444
Trying ::1 ...
Trying 127.0.0.1 ...
Connected to localhost.
Escape character is '^]'.
Open On-Chip Debugger
>
> halt
target halted in Thumb state due to debug-request, current mode: Supervisor
> dump_image dump_proxmark3.bin 0x100000 0x100000
dumped 1048576 bytes in 945.763672s (1.083 KiB/s)
```

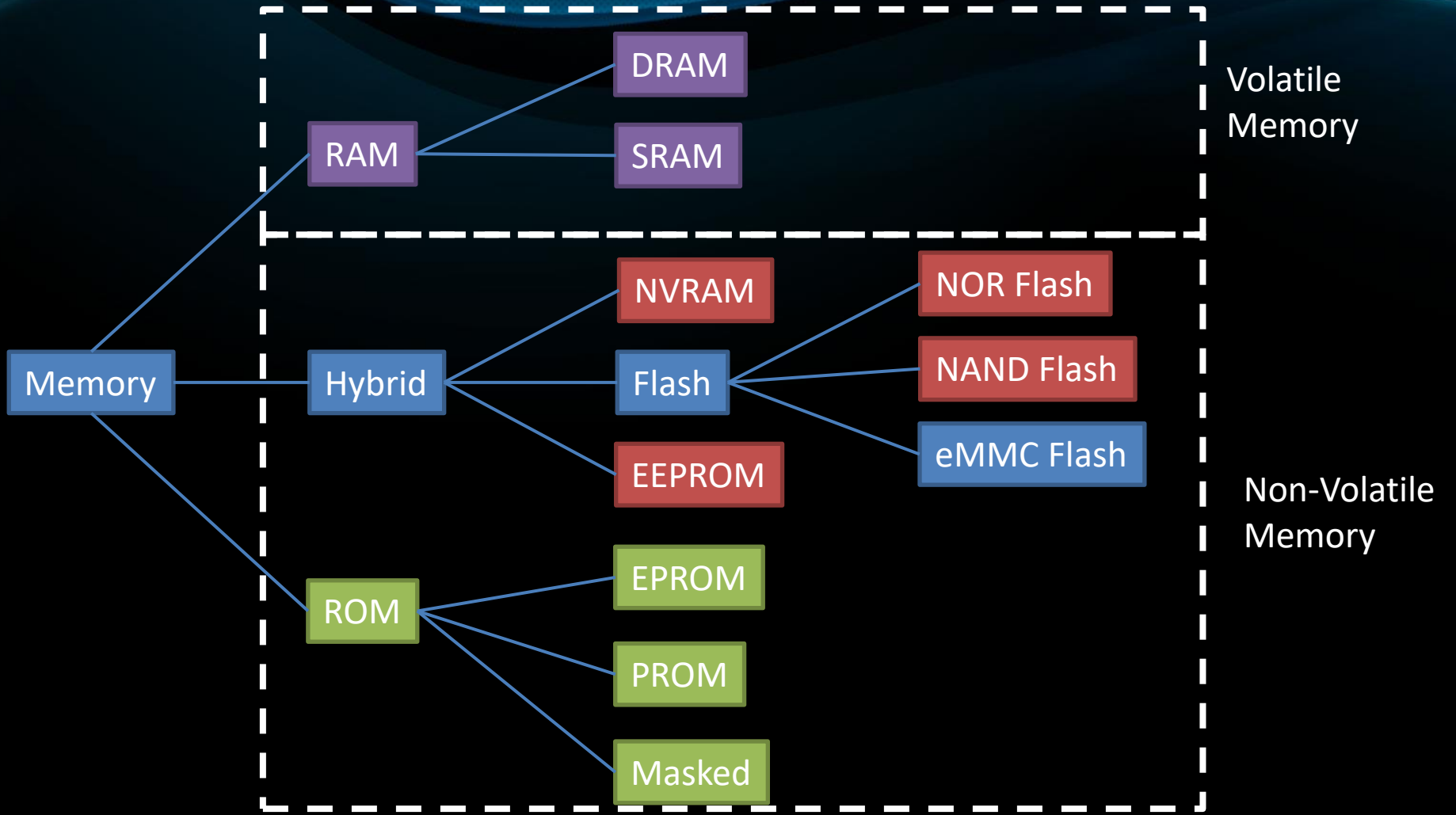
7. Dump contains firmware's ARM code. It is possible to open it with IDA:





6> SPI FLASH

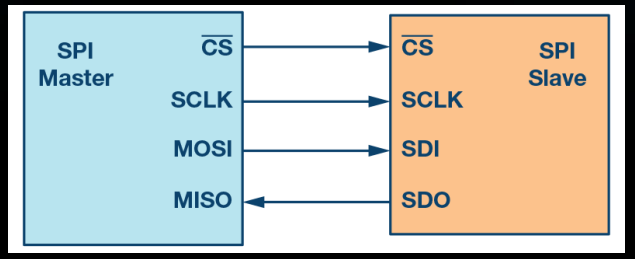
Memory Types



NVRAM, EEPROM, NOR Flash & NAND Flash often use SPI (or I^2C) protocol to communicate with Micro-Controller

SPI (Serial Peripheral Interface) Protocol

- Synchronous (requires clock) serial communication
- Designed for inter-components communication (hi speed)
- 1 Master (Micro-Controller) / Multiple Slaves (eg. Flash)



- SPI Pins:
  - **MISO**: Master In Slave Out (Master <- Slave)
  - **MOSI**: Master Out Slave In (Master -> Slave)
  - **SCLK**: Clock
  - **CS**: Chip Select . Required to select 1 Slave among others for any action. CS put to 0 when chip selected

- Non-standard Pin names are also used. **From Slave point-of-view:**
  - **SDI / DI / DIN / SI:** Data In. Connect to MOSI on Master
  - **SDO / DO / DOUT / SO:** Data Out. Connect to MISO on Master

Identify SPI Flash

- Refer to Flash datasheet to find SPI compatibility & pinout:

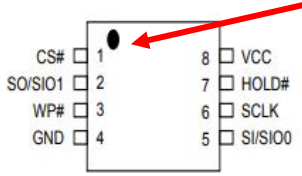
MXIC

MACRONIX INTERNATIONAL CO., LTD.

MX25L3208E

PIN CONFIGURATIONS

8-PIN SOP (200mil)

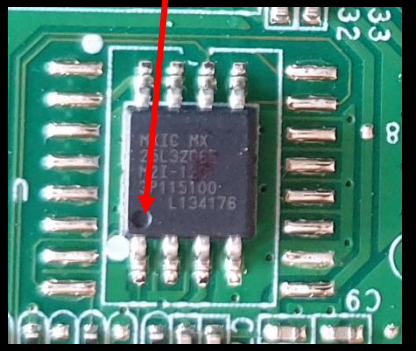


8-LAND WSON (6x5mm)

PIN DESCRIPTION

SYMBOL	DESCRIPTION
CS#	Chip Select
SI/SIO0	Serial Data Input (for 1 x I/O)/ Serial Data Input & Output (for Dual Output mode)
SO/SIO1	Serial Data Output (for 1 x I/O)/ Serial Data Output (for Dual Output mode)
SCLK	Clock Input
WP#	Write protection
HOLD#	Hold, to pause the device without deselecting the device
VCC	+ 3.3V Power Supply
GND	Ground

Circle indicates Pin #1



Serial NOR Flash

GENERAL DESCRIPTION

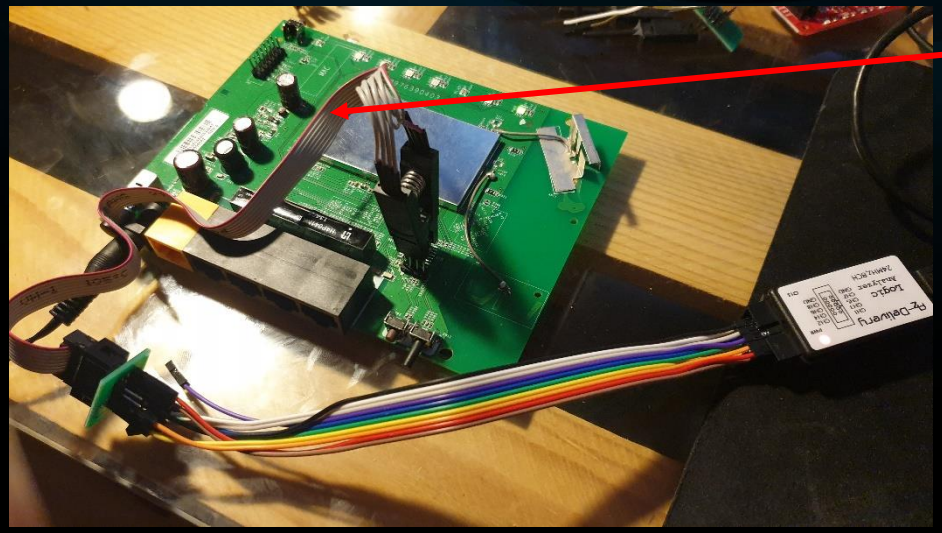
The device feature a serial peripheral interface and software protocol allowing operation on a simple 3-wire bus. The three bus signals are a clock input (SCLK), a serial data input (SI), and a serial data output (SO). Serial access to the device is enabled by CS# input.

When it is in Dual Output read mode, the SI and SO pins become SIO0 and SIO1 pins for data output.

https://pdf1.alldatasheet.com/datasheet-pdf/view/575458/MCNIX/MX25L3208EM2I12G.html

Check SPI Pins with Logic Analyzer

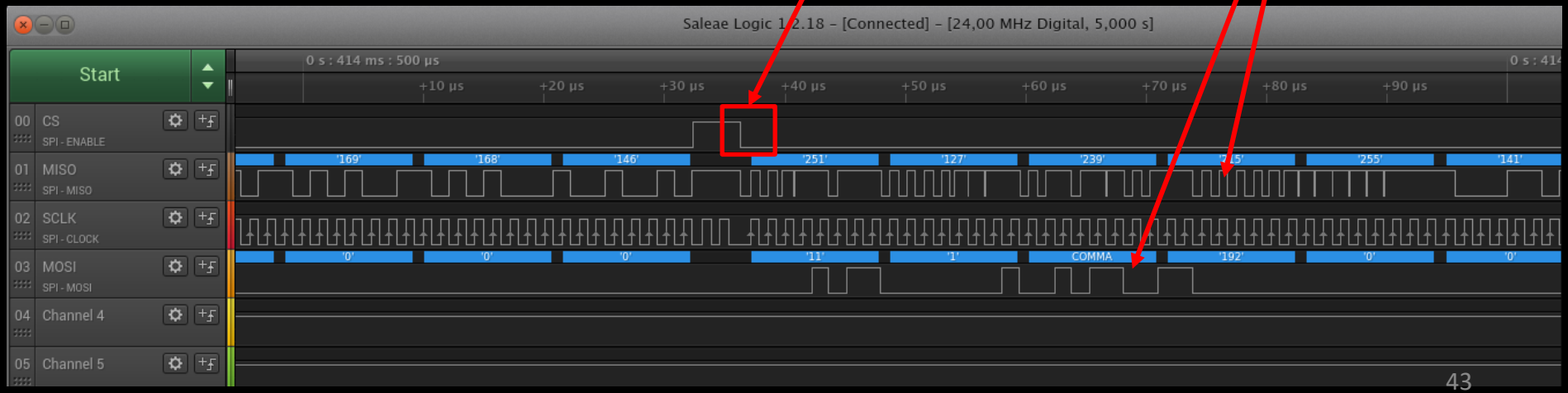
- Connect to the Flash using a SOIC/SOP 8-Pin chip clip with Logic Analyzer



Red cable corresponds to Pin #1

Data emitted/received

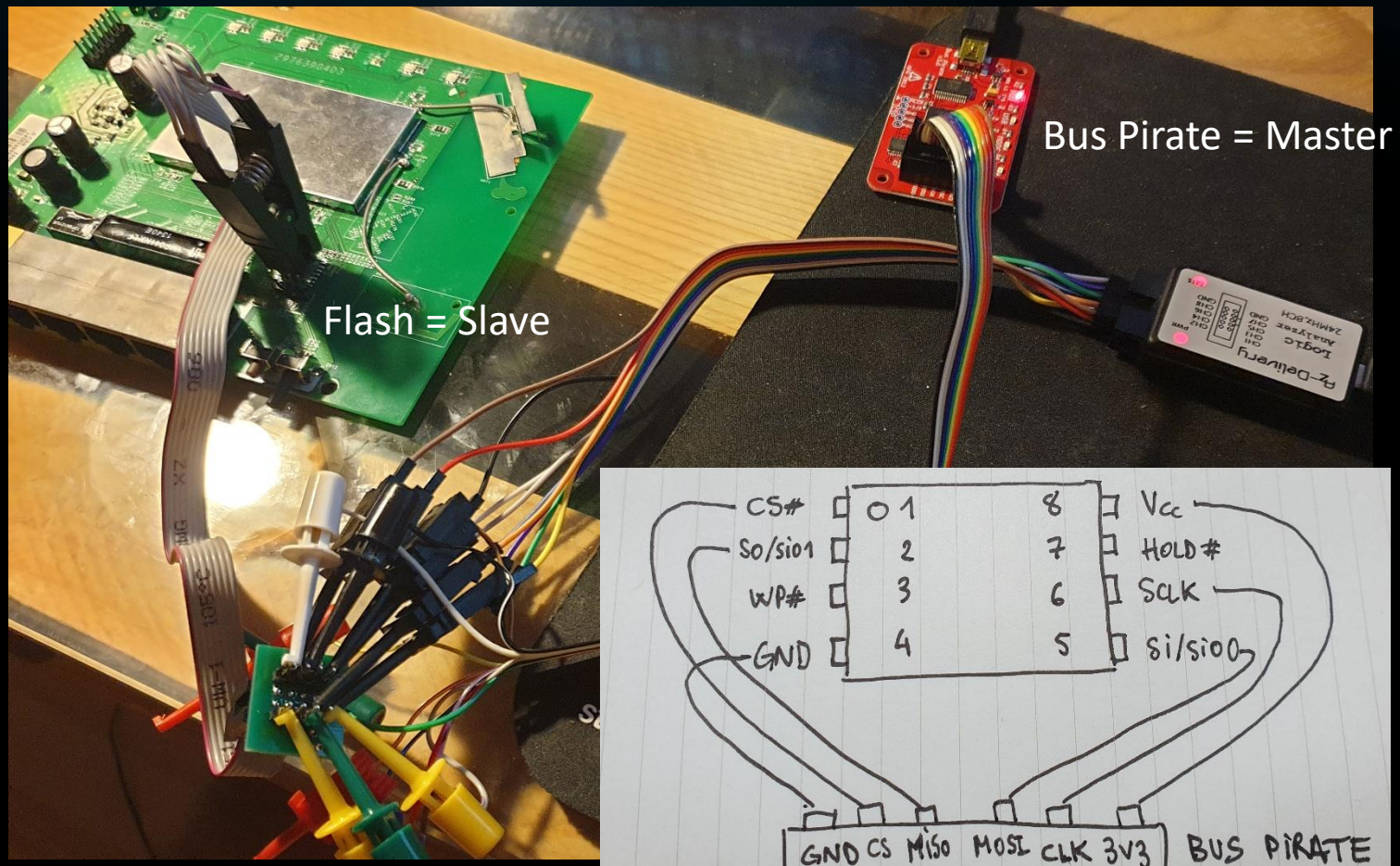
CS (Chip Select) drops to 0 when communication begins



Dump SPI Flash – First try… (1/2)

Now we are sure of Flash pinout, let's try to dump it using Bus Pirate:

1. Connect Bus Pirate as follows:



Dump SPI Flash – First try… (2/2)

2. Check for Flash support in flashrom: https://flashrom.org/Supported_hardware

Macronix	MX25L3206E/MX25L3208E	4096	SPI	OK	OK	OK	OK	2.700	3.600

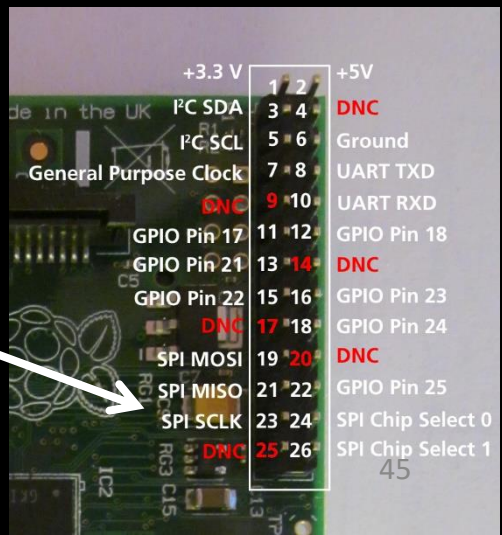
3. Run flashrom, it should auto-detect the Flash (Warning: device powered OFF):

```
root@hackbox:/home/jbr/Projet-Netgear# flashrom -p buspirate_spi:dev=/dev/ttyUSB0
flashrom v1.2 on Linux 5.4.0-kali4-amd64 (x86_64)
flashrom is free software, get the source code at https://flashrom.org

Using clock_gettime for delay loops (clk_id: 1, resolution: 1ns).
Bus Pirate firmware 6.1 and older does not support SPI speeds above 2 MHz. Limiting speed to 2 MHz.
It is recommended to upgrade to firmware 6.2 or newer.
No EEPROM/flash device found.
Note: flashrom can never write if the flash chip isn't found automatically.
```

FAIL !

4. Try to find cause of fail:
   - "Sniff" communication with Logic Analyzer
   - Also, test with Raspberry Pi as SPI Interface

=> Unfortunately, seems like there are interferences with other components on the board

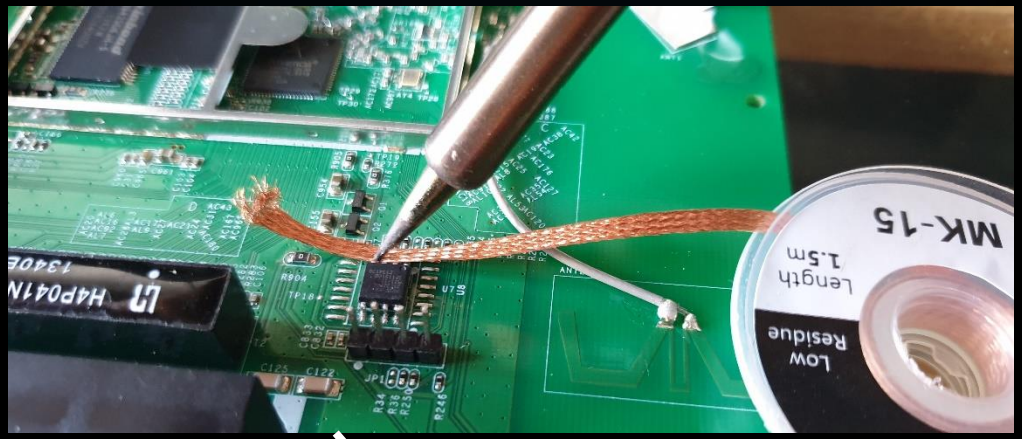


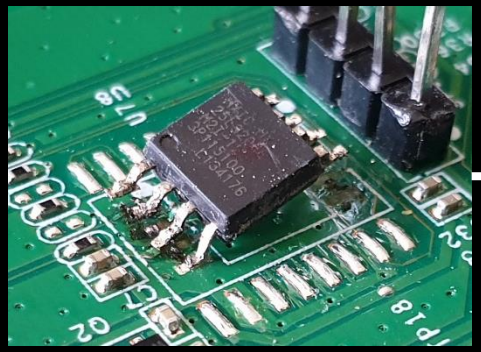
Let’s get serious

Removing Flash from PCB

- When everything else has failed => Desolder chip from PCB
- Very invasive, can be very hard to solder it back !! => Possible PCB destruction !
- Recommended tool: Hot air gun

- I don’t have one, let’s do it the dirty way ☺

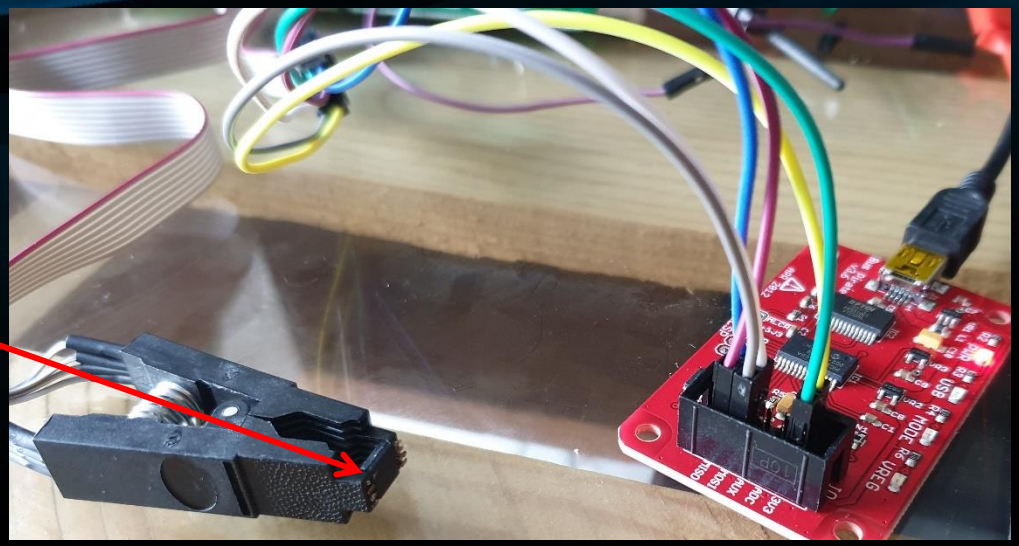




Dump SPI Flash – Second try

1. Connect to Bus Pirate again:

Flash is in the clip



2. Run flashrom again:

```
root@hackbox:/home/jbr/Projet-Netgear# flashrom -p buspirate_spi:dev=/dev/ttyUSB0
flashrom v1.2 on Linux 5.4.0-kali4-amd64 (x86_64)
flashrom is free software, get the source code at https://flashrom.org

Using clock_gettime for delay loops (clk_id: 1, resolution: 1ns).
Bus Pirate firmware 6.1 and older does not support SPI speeds above 2 MHz. Limiting speed to 2 MHz.
It is recommended to upgrade to firmware 6.2 or newer.
Found Macronix flash chip "MX25L3206E/MX25L3208E" (4096 kB, SPI) on buspirate_spi.
```

Matching correct Flash

3. Dump Flash memory content:

```
root@hackbox:/home/jbr/Projet-Netgear# flashrom -p buspirate_spi:dev=/dev/ttyUSB0,spispeed=1M -c "MX25L3206E/MX25L3208E" -r dump.bin
flashrom v1.2 on Linux 5.4.0-kali4-amd64 (x86_64)
flashrom is free software, get the source code at https://flashrom.org

Using clock_gettime for delay loops (clk_id: 1, resolution: 1ns).
Found Macronix flash chip "MX25L3206E/MX25L3208E" (4096 kB, SPI) on buspirate_spi.
Reading flash ... done.
root@hackbox:/home/jbr/Projet-Netgear#
```

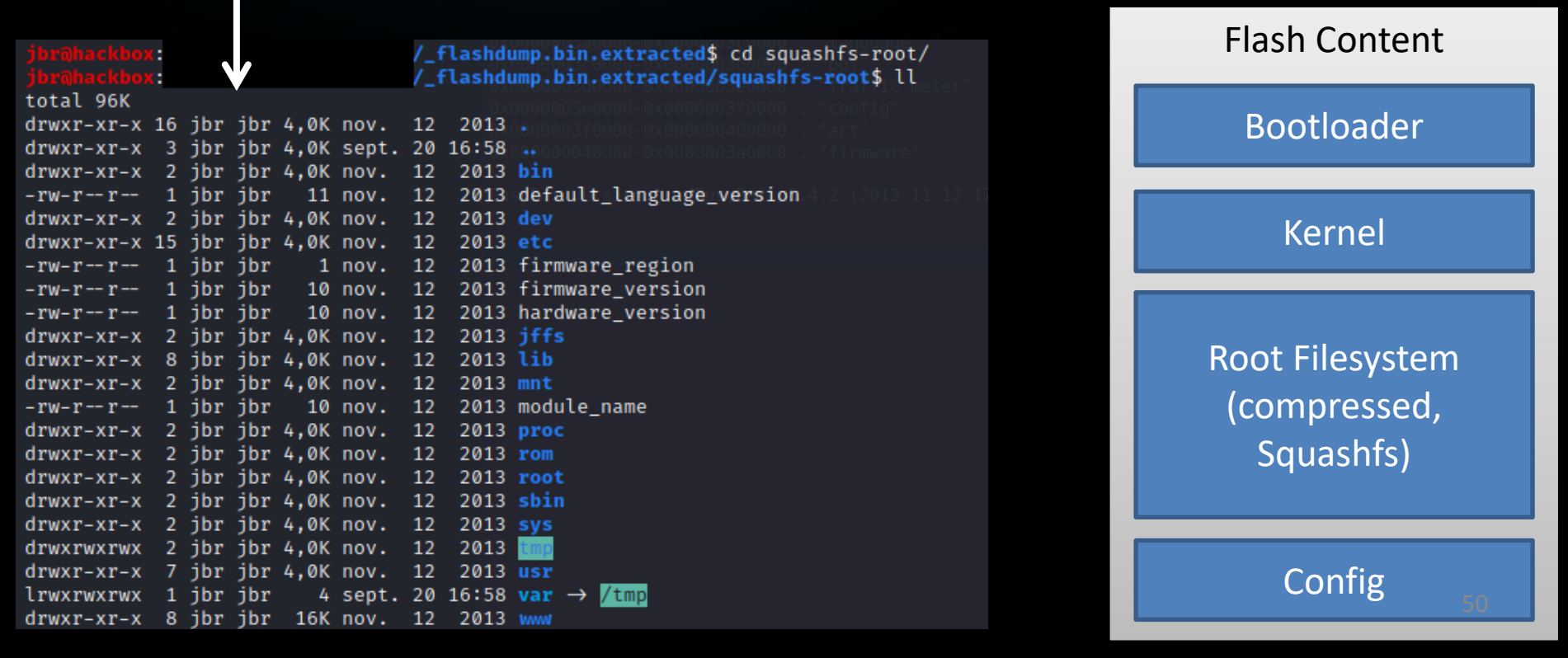
7> FIRMWARE ANALYSIS

Root Filesystem Extraction

- Automatic Filesystem extraction with binwalk:

```
jbr@hackbox:                    $ binwalk -e flashdump.bin

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
134816        0x20EA0         Certificate in DER format (x509 v3), header length: 4, sequence length: 64
150864        0x24D50         U-Boot version string, "U-Boot 1.1.4 (Nov 26 2012 - 15:58:42)"
151232        0x24EC0         CRC32 polynomial table, big endian
160905        0x27489         Copyright string: "copyright."
262208        0x40040         LZMA compressed data, properties: 0x6D, dictionary size: 8388608 bytes, uncompressed size: 2465316 bytes
1114112       0x110000        Squashfs filesystem, little endian, version 4.0, compression:lzma, size: 2676149 bytes, 1117 inodes, blocksize: 131072 bytes,
3801092       0x3A0004        POSIX tar archive (GNU), owner user name: "_table.tar.gz"
```

```
jbr@hackbox:                    /_flashdump.bin.extracted$ cd squashfs-root/
jbr@hackbox:                    /_flashdump.bin.extracted/squashfs-root$ ll
total 96K
drwxr-xr-x 16 jbr jbr 4,0K nov.  12  2013 .
drwxr-xr-x  3 jbr jbr 4,0K sept. 20 16:58 ..
drwxr-xr-x  2 jbr jbr 4,0K nov.  12  2013 bin
-rw-r--r--  1 jbr jbr   11 nov.  12  2013 default_language_version
drwxr-xr-x  2 jbr jbr 4,0K nov.  12  2013 dev
drwxr-xr-x 15 jbr jbr 4,0K nov.  12  2013 etc
-rw-r--r--  1 jbr jbr    1 nov.  12  2013 firmware_region
-rw-r--r--  1 jbr jbr   10 nov.  12  2013 firmware_version
-rw-r--r--  1 jbr jbr   10 nov.  12  2013 hardware_version
drwxr-xr-x  2 jbr jbr 4,0K nov.  12  2013 jffs
drwxr-xr-x  8 jbr jbr 4,0K nov.  12  2013 lib
drwxr-xr-x  2 jbr jbr 4,0K nov.  12  2013 mnt
-rw-r--r--  1 jbr jbr   10 nov.  12  2013 module_name
drwxr-xr-x  2 jbr jbr 4,0K nov.  12  2013 proc
drwxr-xr-x  2 jbr jbr 4,0K nov.  12  2013 rom
drwxr-xr-x  2 jbr jbr 4,0K nov.  12  2013 root
drwxr-xr-x  2 jbr jbr 4,0K nov.  12  2013 sbin
drwxr-xr-x  2 jbr jbr 4,0K nov.  12  2013 sys
drwxrwxrwx  2 jbr jbr 4,0K nov.  12  2013 tmp
drwxr-xr-x  7 jbr jbr 4,0K nov.  12  2013 usr
lrwxrwxrwx  1 jbr jbr    4 sept. 20 16:58 var → /tmp
drwxr-xr-x  8 jbr jbr  16K nov.  12  2013 www
```



Firmware Analysis

- Static Analysis:
  - Explore filesystem: Search for interesting files, configurations… (firmwalker)
  - Check various scripts
  - Binary analysis (IDA)

- Dynamic Analysis:
  - Emulate firmware with firmadyne tool (https://github.com/firmadyne/firmadyne) + QEMU (support for MIPS & ARM)
  - Binary exploitation often easier because usually less defense mechanisms (ASLR, NX…) on embedded devices

At this step, we deal with stuff we are more familiar with !