KU LEUVEN

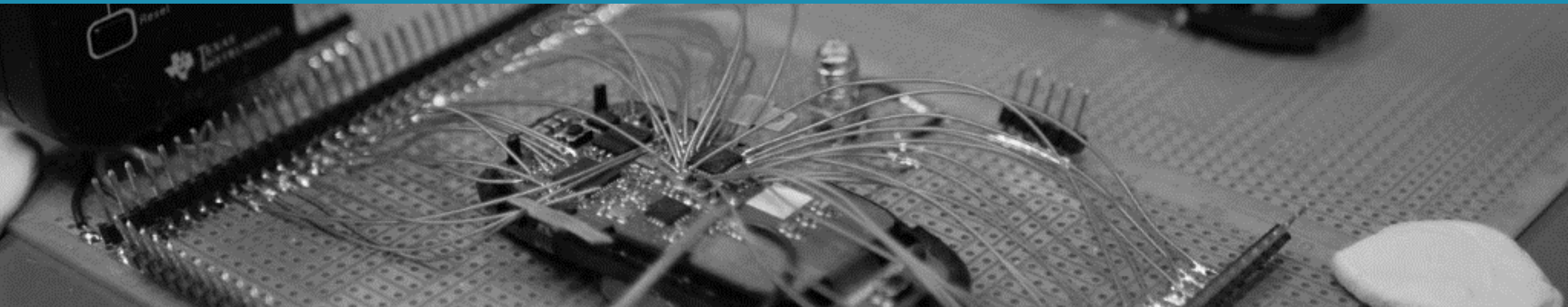
Security of embedded systems

an (offensive) introduction

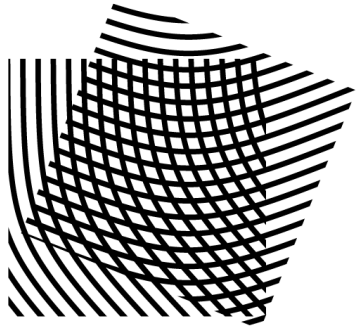
COSIC

Lennert Wouters

SecAppDev 2022



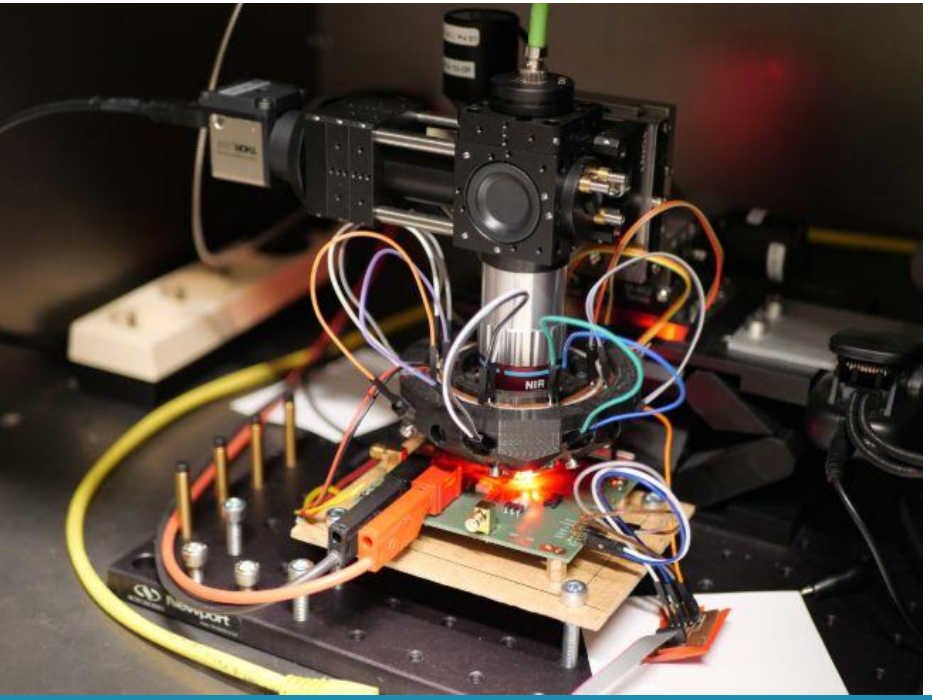
- Symmetric Key Cryptography
- Public Key and Protocols Team
- Privacy and Identity Management
- Mobile and Wireless Security
- **Embedded Systems and hardware Security Team**

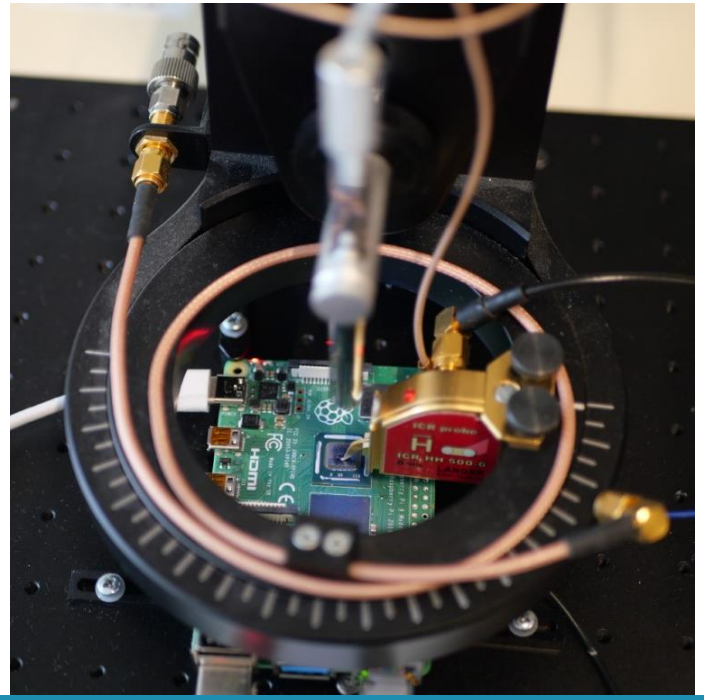


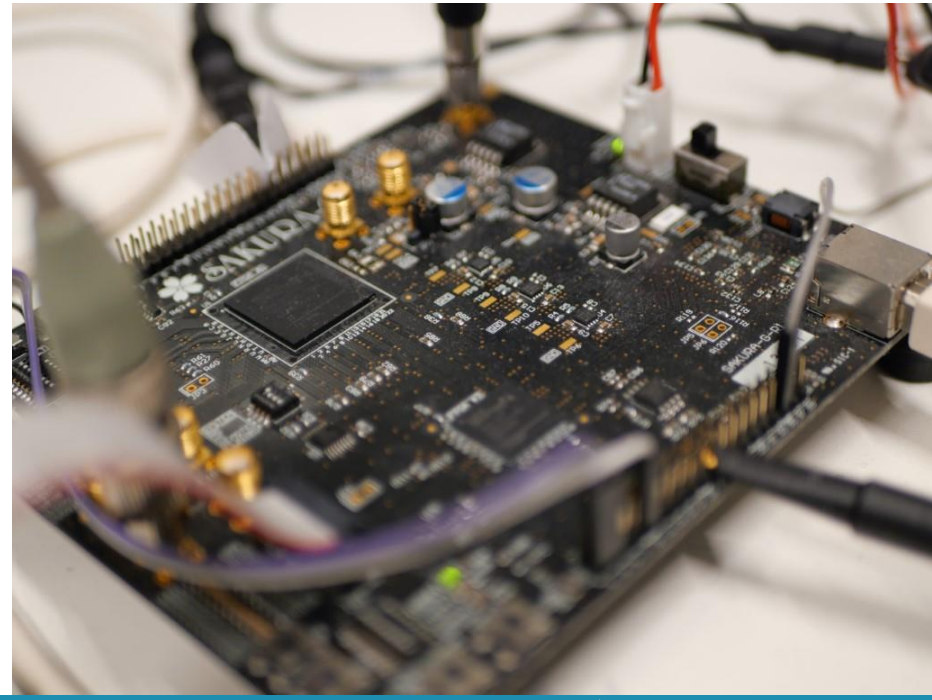
THE WALL STREET JOURNAL.

In Belgium, an Encryption Powerhouse Rises

University of Leuven has become a battleground in the fight between privacy and surveillance







COSIC

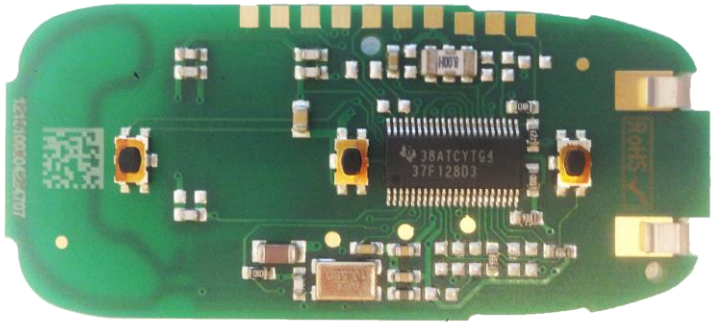
KU LEUVEN

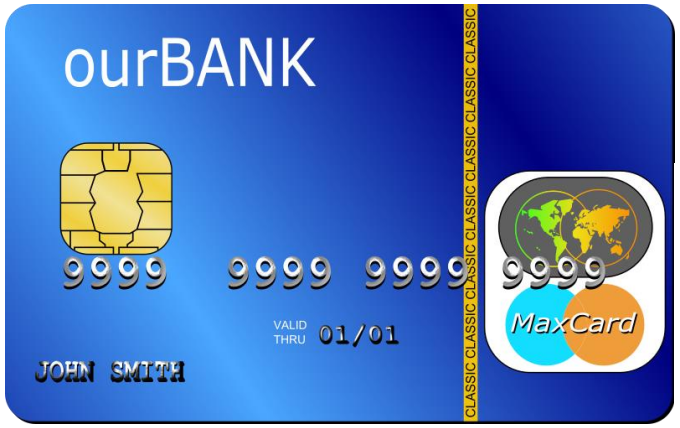
Outline

- Introduction
- Threat modeling
- Embedded attackers and their capabilities
- Case studies
  - Car keys
  - Smart devices

Embedded devices

- Processing unit
  - 8-bit 8051 → multi-core CPU
- Volatile memory
  - DRAM, SRAM
- Non-volatile memory
  - ROM, FLASH
    - Baremetal
    - Real-Time OS
    - Rich OS
- Communication interface(s)





COSIC

KU LEUVEN

Ecosystem

Further reading: https://www.pentestpartners.com/security-blog/vulnerable-wi-fi-dildo-camera-endoscope-yes-really/
https://www.pentestpartners.com/security-blog/smart-male-chastity-lock-cock-up/

https://arstechnica.com/tech-policy/2014/05/photos-of-an-nsa-upgrade-factory-show-cisco-router-getting-implant/

Threat modeling

- What are we trying to protect?
  - Assets and security objectives
- Who are we are trying to protect from?
  - The attacker model
    - Goals, resources and capabilities
- The cost of an attack should outweigh the benefit
- Take the entire device lifecycle into consideration!
  - Supply chain
  - Manufacturing
  - Shipping
  - Product returns
  - Device lifetime

(TS//SI//NF) Left: Intercepted packages are opened carefully; Right: A "load station" implants a beacon

COSIC

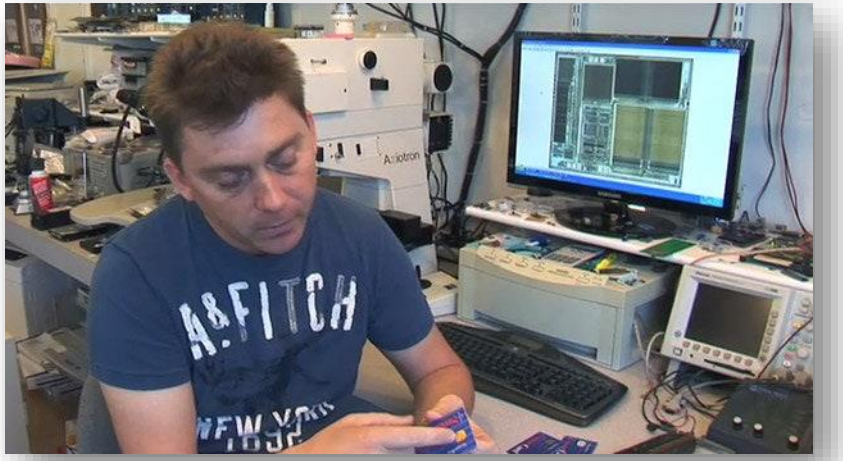
KU LEUVEN

Threat modeling: assets and objectives

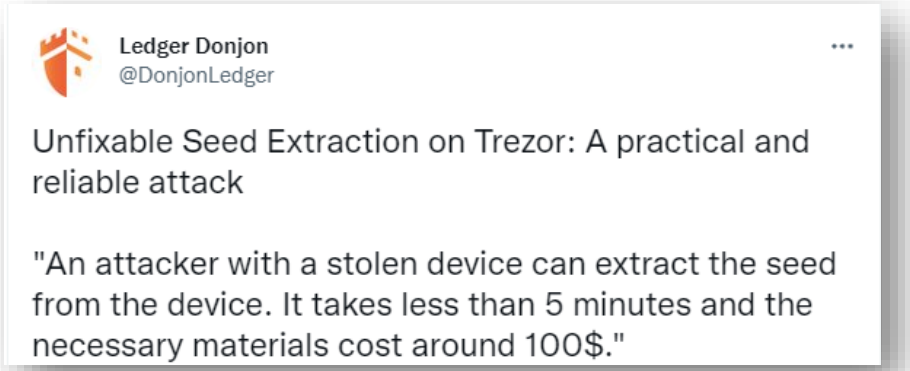
- Confidentiality and integrity
  - Binary code
    - Integrity: prevent modification ←→ can prevent you from running your own code
    - Confidentiality: prevent copying intellectual property, more difficult to find vulnerabilities
  - Key material
    - Integrity: prevent modification of a root-of-trust public key
    - Confidentiality: is linked to confidentiality of data
  - Personally identifiable information (PII)
  - Sensor data
    - Can be related to PII
- Other objectives (e.g. safety) can be at odds with security

Threat modeling: attackers and their goals

- Researchers
  - Bug bounty, publication, fame
- Criminal enterprise
  - Financial gain → scalable attack
- Industry competition
  - “competitive analysis”
  - IP infringement / damaging reputation
- Nation-state
  - Sabotage, espionage, counterterrorism



https://archive.org/details/MurdochsTVPirates
https://www.wired.com/2008/05/tarnovsky/



https://twitter.com/DonjonLedger/status/1146323810167197696

COSIC KU LEUVEN

Threat modeling: cost versus benefit

- Cost
  - Money, time, frustration
  - How will this cost evolve over the device lifetime?
- Benefit
  - Money, goods
  - PII, paper, fame
- Scalability
  - Cost of attack discovery vs cost of attack exploitation
  - Break Once Run Everywhere (BORE)
    - PayTV smartcards
    - Game console modchips

Threat modeling: consumer viewpoint

- Are you worried about a random hacker, an ex-partner or a nation-state?
- Can you really trust the hardware you buy?
  - Do you trust the product designer?
  - Do you trust the fabrication process?
  - What about the hardware you received?

Forbes

CYBERSECURITY

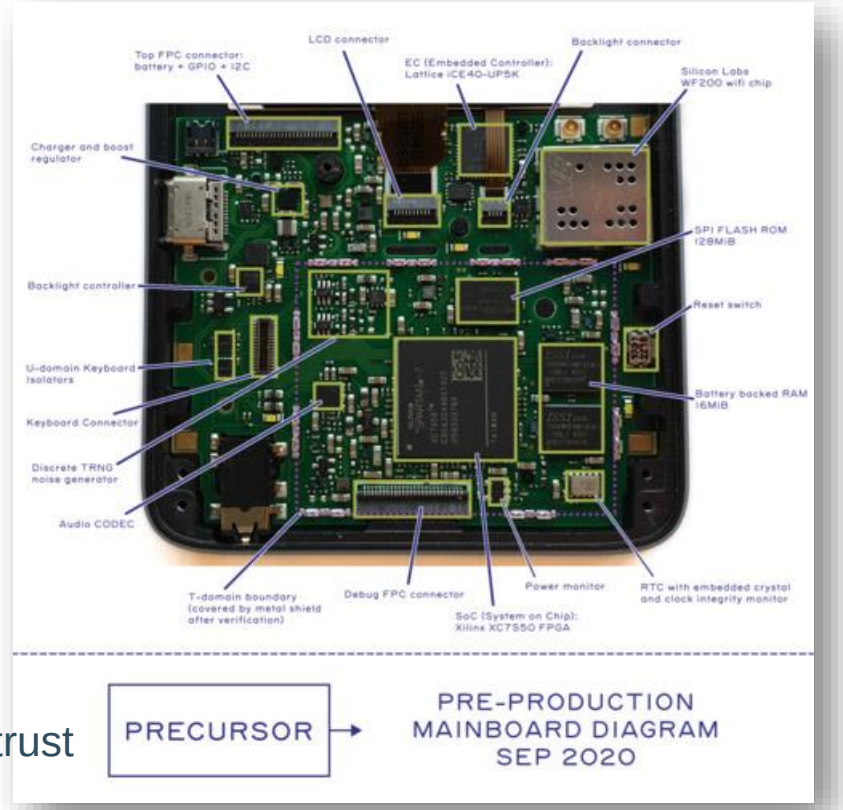
Cybercrooks Are Mailing Users Fake Ledger Devices To Steal Their Cryptocurrency

Lee Mathews Senior Contributor ⓘ
Observing, pondering, and writing about tech. Generally in that order.

Follow

Jun 18, 2021, 03:46pm EDT

Ledger's hardware wallets are a very popular way to securely store cryptocurrency. So popular, in fact, that cybercriminals have cooked up an elaborate scam that targets Ledger users by mailing them bogus replacement wallets.



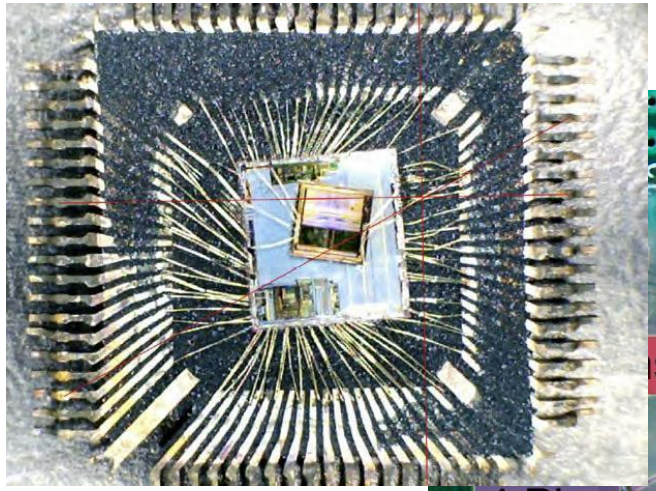
Precursor – evidence based trust
Bunnie Huang
https://www.youtube.com/watch?v=w8BA6_9HCzk

https://www.forbes.com/sites/leemathews/2021/06/18/cybercrooks-are-mailing-users-fake-ledger-devices-to-steal-their-cryptocurrency

COSIC KU LEUVEN

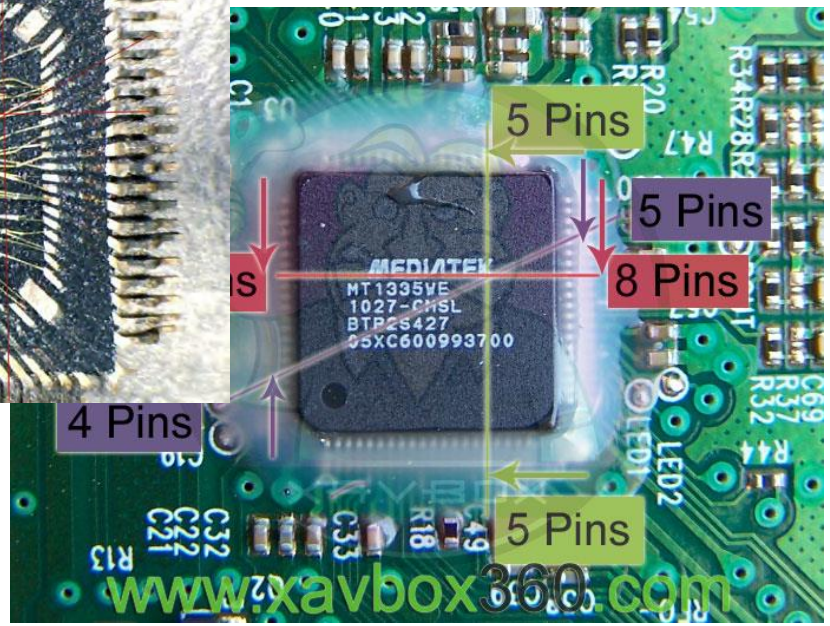
Hardware threat modeling: key differences

- Many embedded devices are exposed to a hostile environment containing adversaries that have physical access to the device
- In some scenarios the (physical) attacks are conducted by the legitimate user!

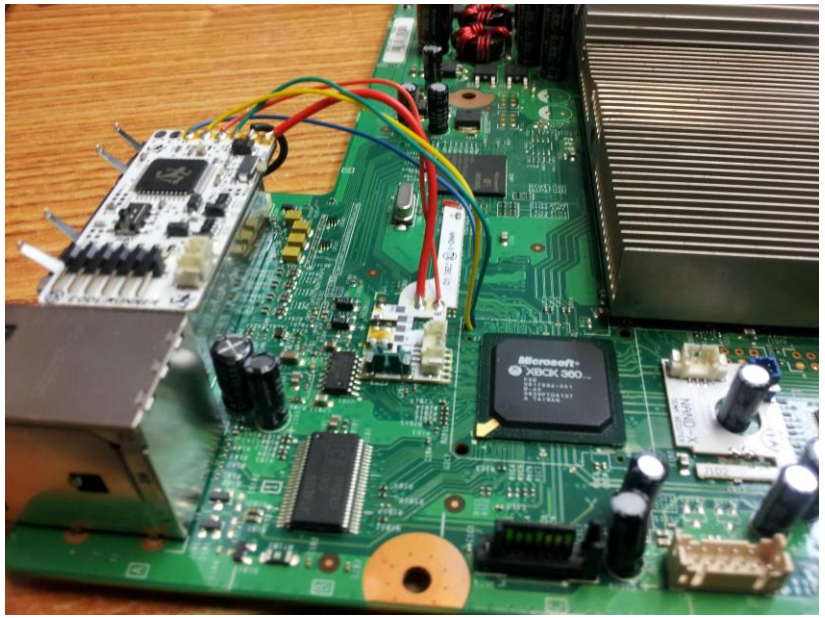


Source:
Xavbox
Team-xecuter

Kamikaze attack



Reset Glitch Hack



Source:
undersyde.wordpress.com

Guarding Against Physical Attacks: The Xbox One Story – Tony Chen, Microsoft
https://www.platformsecuritysummit.com/2019/speaker/chen/

Embedded attacker capabilities: outline

- Basic device exploration
  - Accessing debug interfaces
  - Techniques for extracting firmware from a device
    - (assuming it is not available for download)

Can often be performed at home with basic equipment!

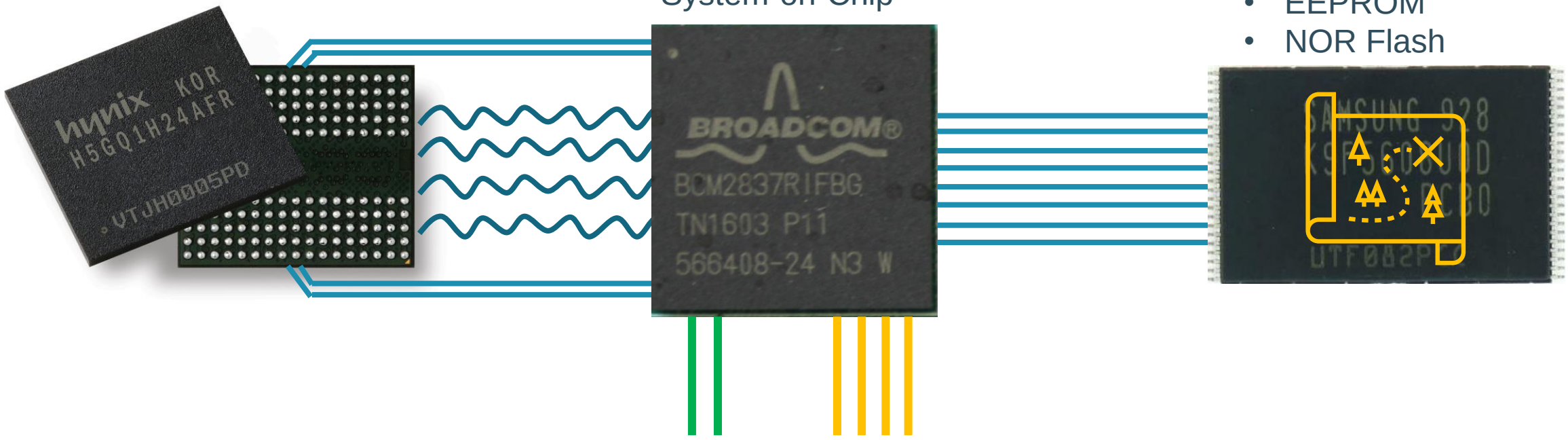
- When faced with countermeasures (e.g. verified boot, firmware encryption)
  - Chip level physical attacks
    - Non-invasive, semi-invasive and invasive

Embedded devices

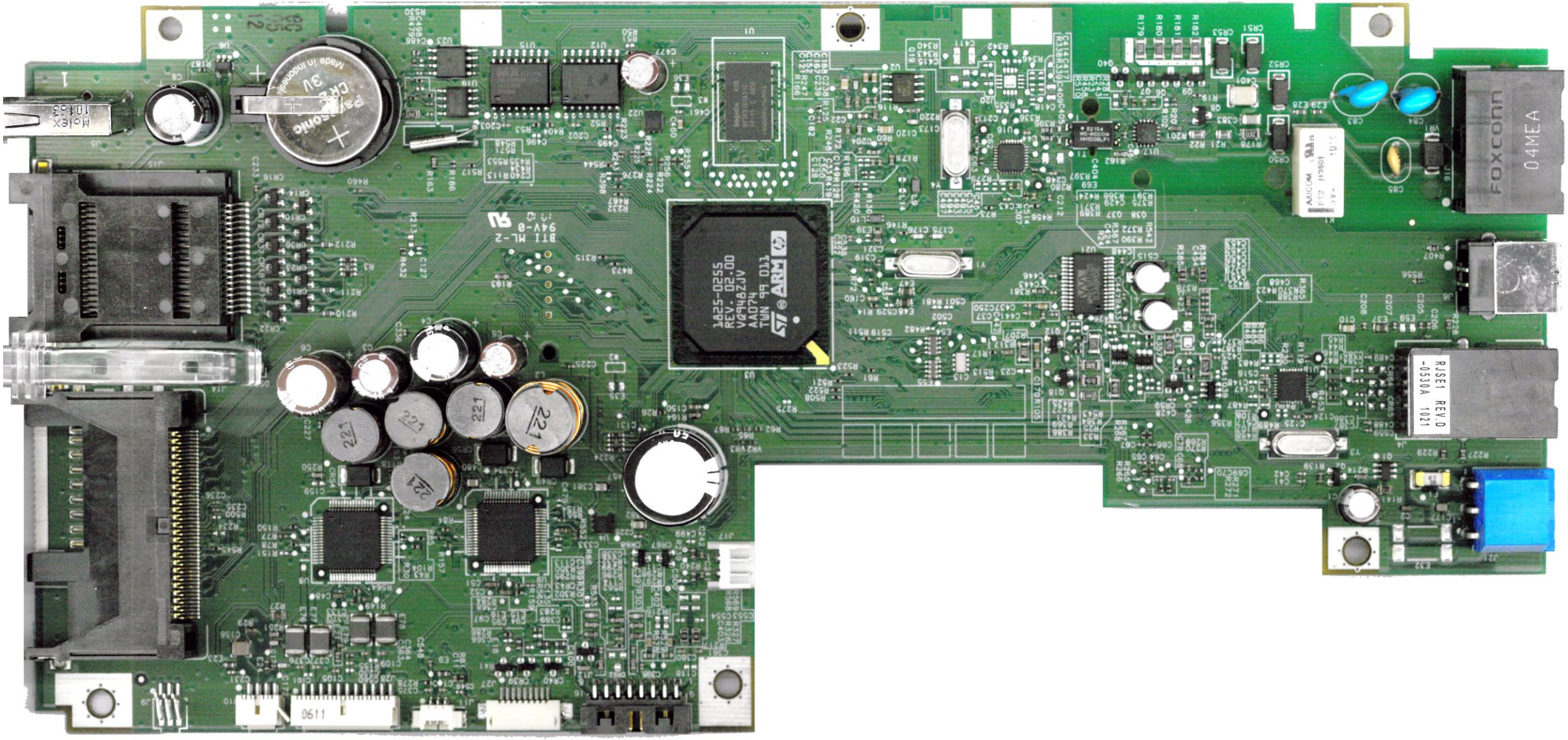
Volatile memory (DRAM)

System-on-Chip

Non-volatile memory
- NAND Flash
- eMMC
- EEPROM
- NOR Flash

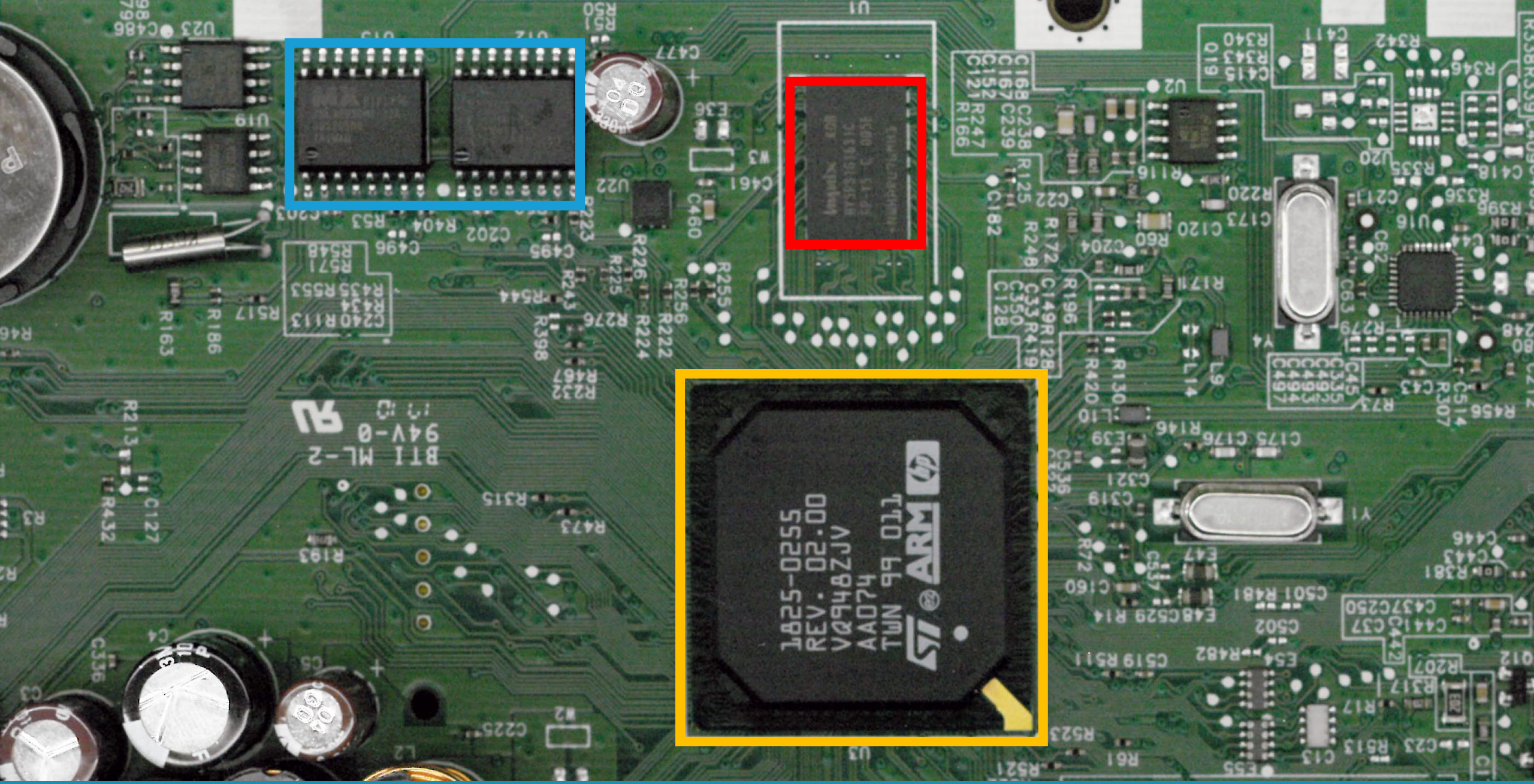


Debug interfaces



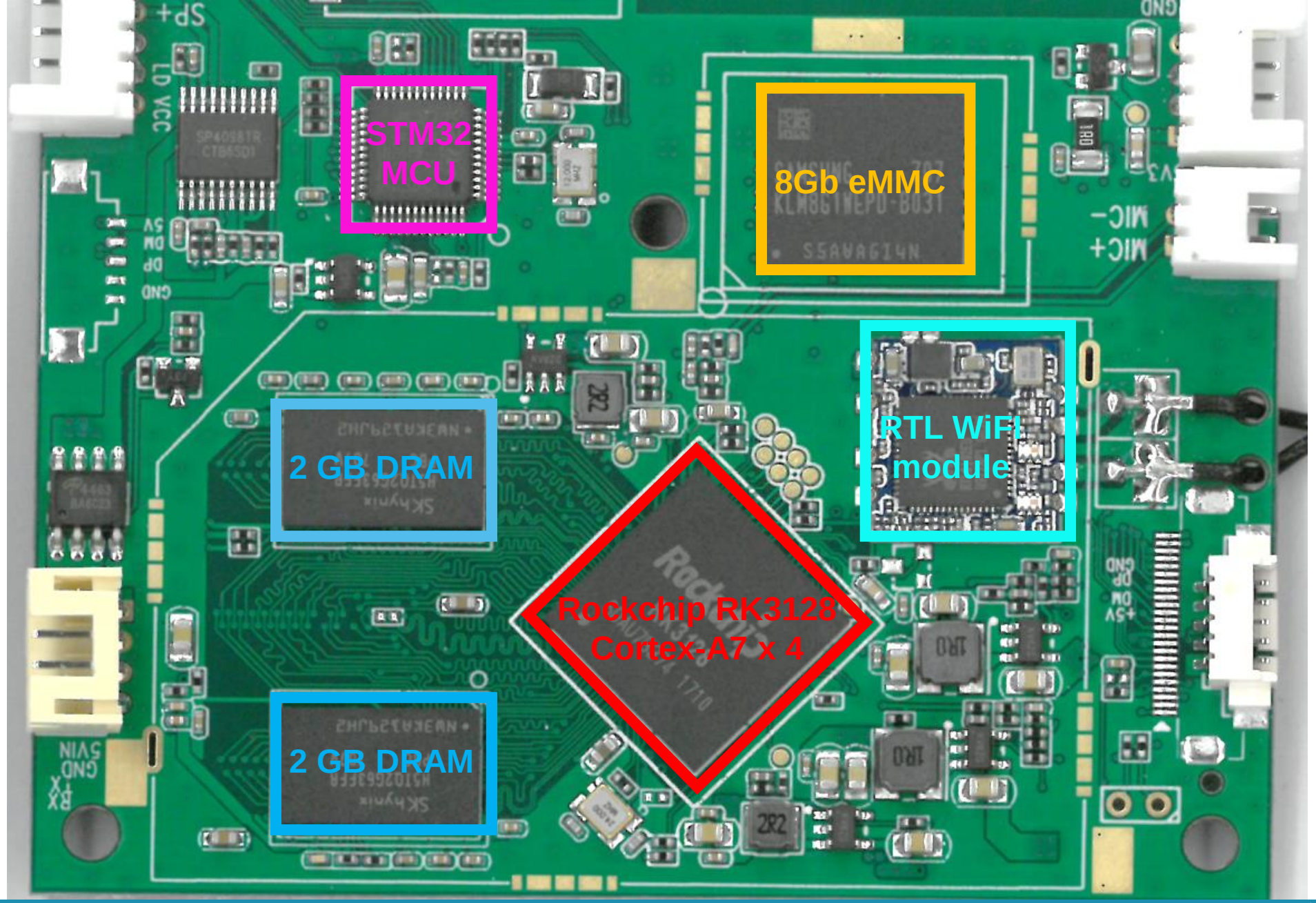
COSIC

KU LEUVEN



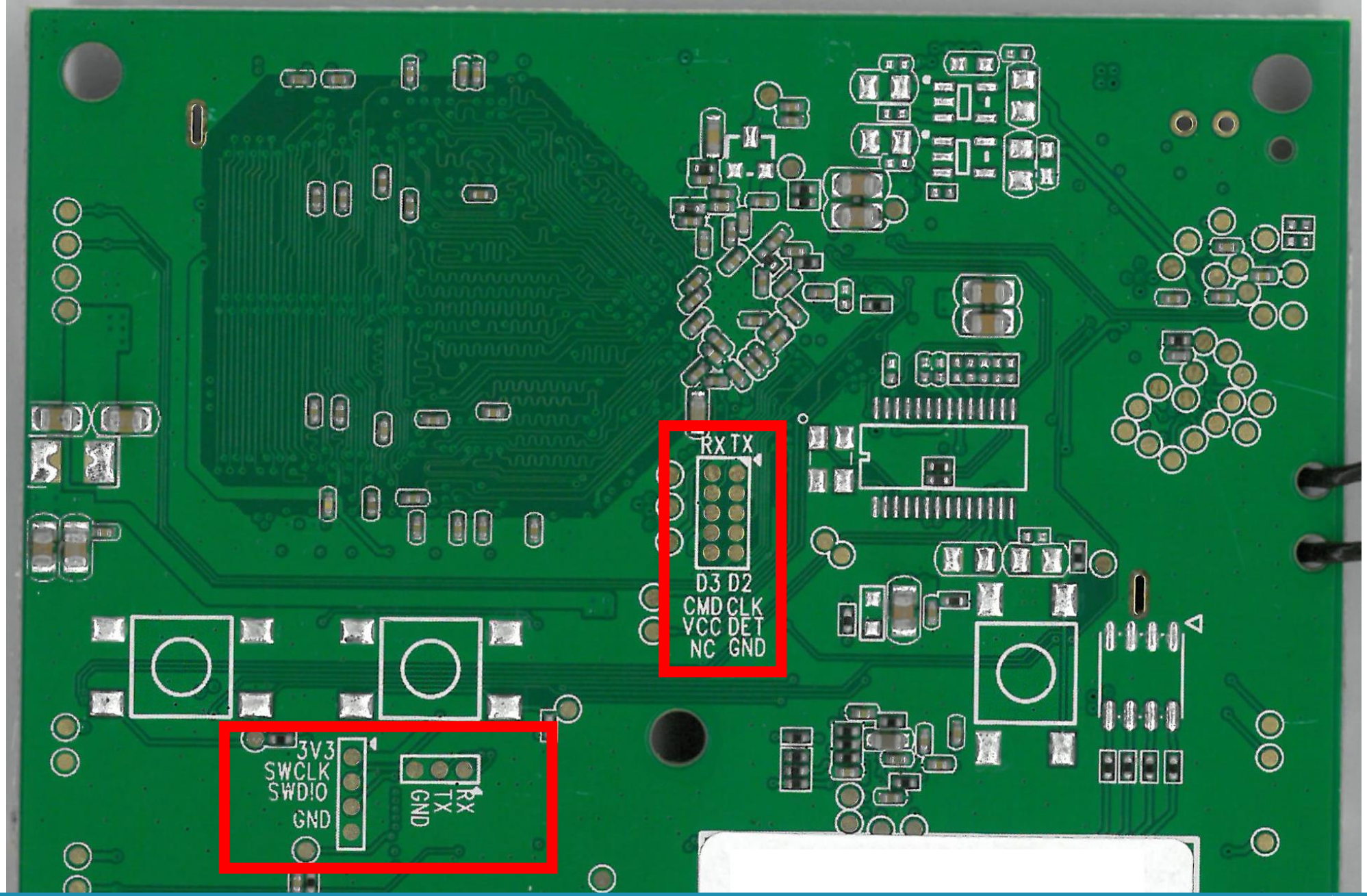
COSIC

KU LEUVEN



COSIC

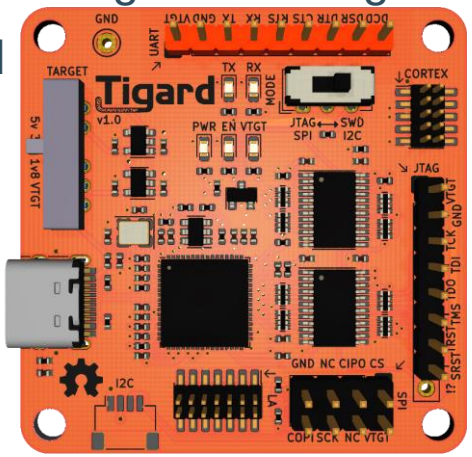
KU LEUVEN


RX TX
D3 D2
CMD CLK
VCC DET
NC GND
3V3
SWCLK
SWDIO
GND
RX
TX
GND

https://github.com/tigard-tools/tigard

Discovering interfaces

Tigard



- UART, SWD, JTAG, SPI, I2C, eMMC, USB, PCIe, …
- Varying levels of complexity
  - Some basic knowledge is helpful during identification
  - Most of the time you can get away with using the right (multi-)tool!

- Connect to test points
  - Solder wires to the board
  - Probe individual test points
  - (restart the device)

Sensepeek PCBite
https://sensepeek.com/

Saleae Logic Analyser
https://www.saleae.com/

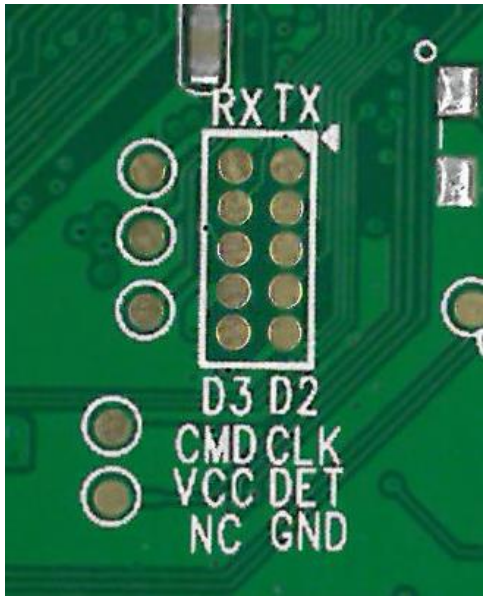
Sigrok
https://sigrok.org/

COSIC

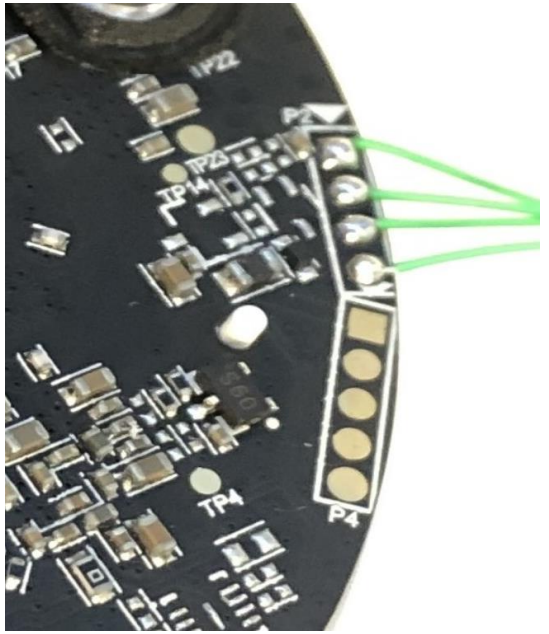
KU LEUVEN

Discovering interfaces: UART

- Universal Asynchronous Receiver-Transmitter
- GND – RX – TX – (VCC)



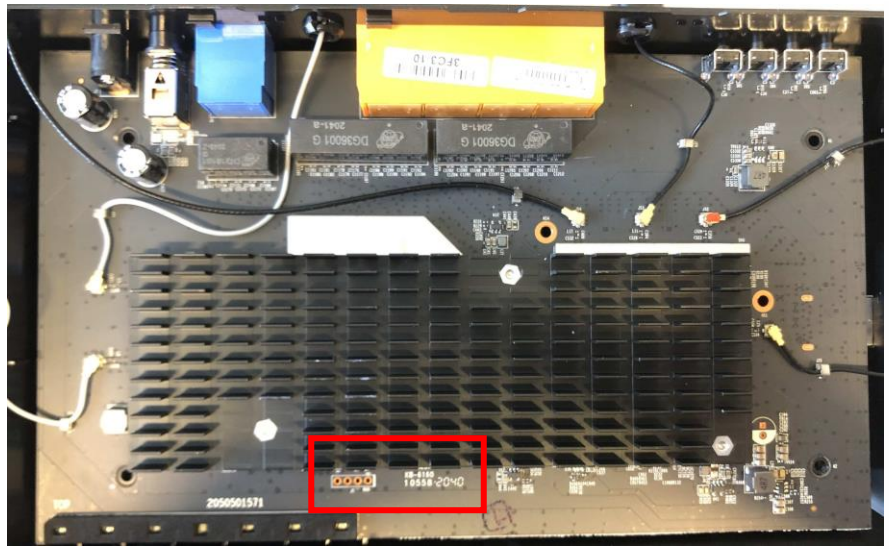
- Can be found on many devices
- Often provides very useful output during boot
  - Occasionally you can access a bootloader CLI
- Occasionally provides a root shell!



Further reading: http://www.devttys0.com/2012/11/reverse-engineering-serial-ports/

Discovering interfaces: UART

TP-Link Archer AX73



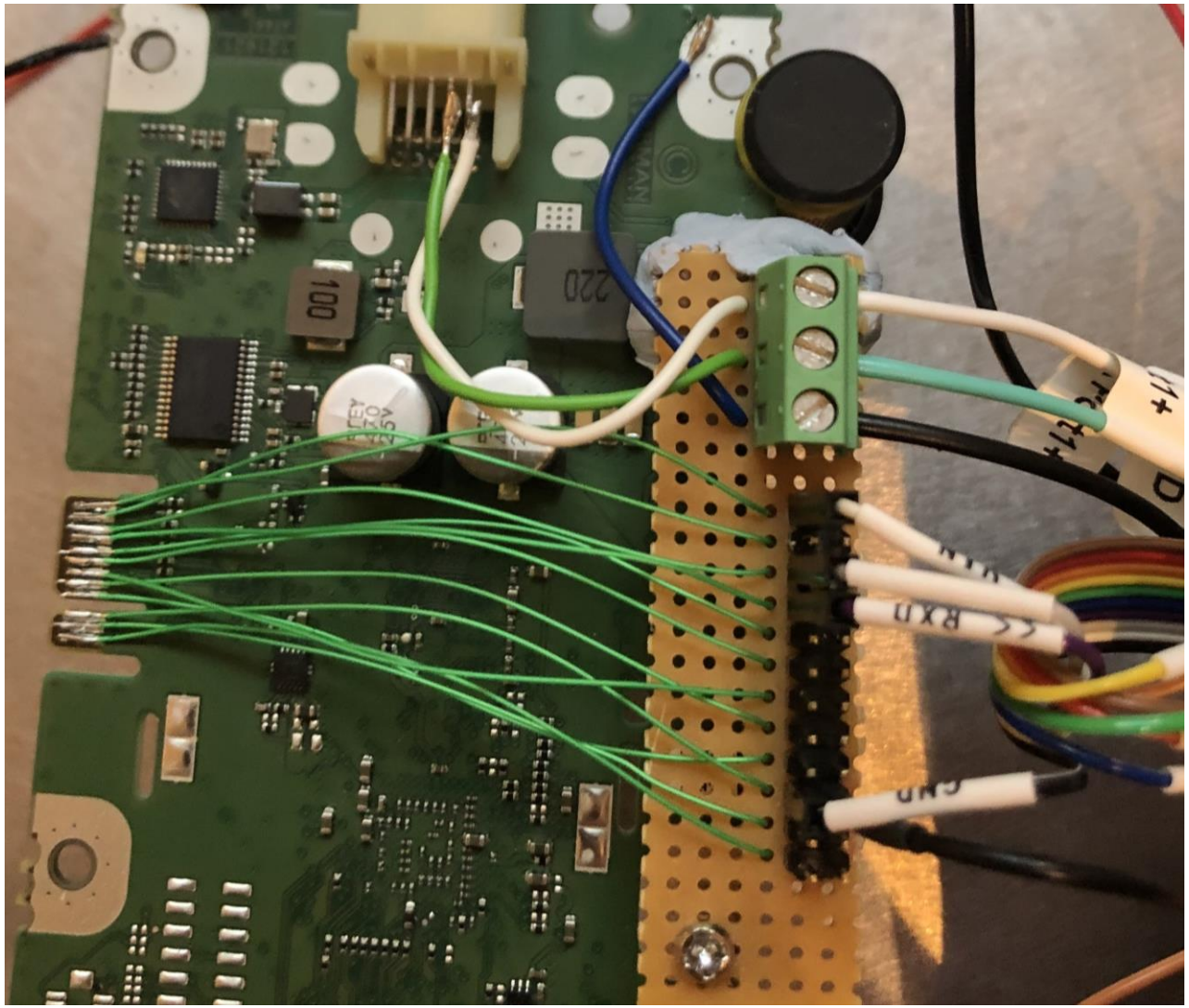

```
BusyBox v1.19.4 (2020-08-26 22:08:11 CST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

     MM          NM                      MMMMMMM           M        M
   $MMMMM        MMMMM                 MMMMMMMMMMM       MMM      MMM
  MMMMMMMM      MM MMMM.               MMMMM:MMMMMM:     MMMM    MMMMM
MMMM= MMMMMM  MMM   MMMM       MMMMM   MMMM  MMMMMM    MMMM   MMMMM'
MMMM=  MMMMM MMMM    MM       MMMMM    MMMM    MMMM    MMMMNMMMMM
MMMM=   MMMM  MMMM             MMMMM    MMMM    MMMM    MMMMMMMM
MMMM=   MMMM   MMMMM           MMMMM    MMMM    MMMM    MMMMMMMM
MMMM=   MMMM     MMMMM,     NMMMMMMMM   MMMM    MMMM    MMMMMMMMM
MMMM=   MMMM      MMMMM   MMMMMMMM     MMMM    MMMM    MMMM  MMMMM
MMMM=   MMMM   MM    MMMM    MMMM      MMMM    MMMM    MMMM    MMMM
MMMM$ ,MMMMM  MMMM  MMMM    MMM       MMMM   MMMMM    MMMM    MMMM
  MMMMMM:      MMMMMM     M          MMMMMMMMMMM  MMMMMMM MMMMMM
    MMMMM       MMMMN     M          MMMMMMMM      MMMM    MMMM
     MMM         M                    MMMMMM       M       M
      M

 ----------------------------------------------------------------
   For those about to rock... (Attitude Adjustment, unknown)
 ----------------------------------------------------------------
root@Archer_AX73:/# id
uid=0(root) gid=0(root)
root@Archer_AX73:/#
```


COSIC KU LEUVEN

Discovering interfaces: UART

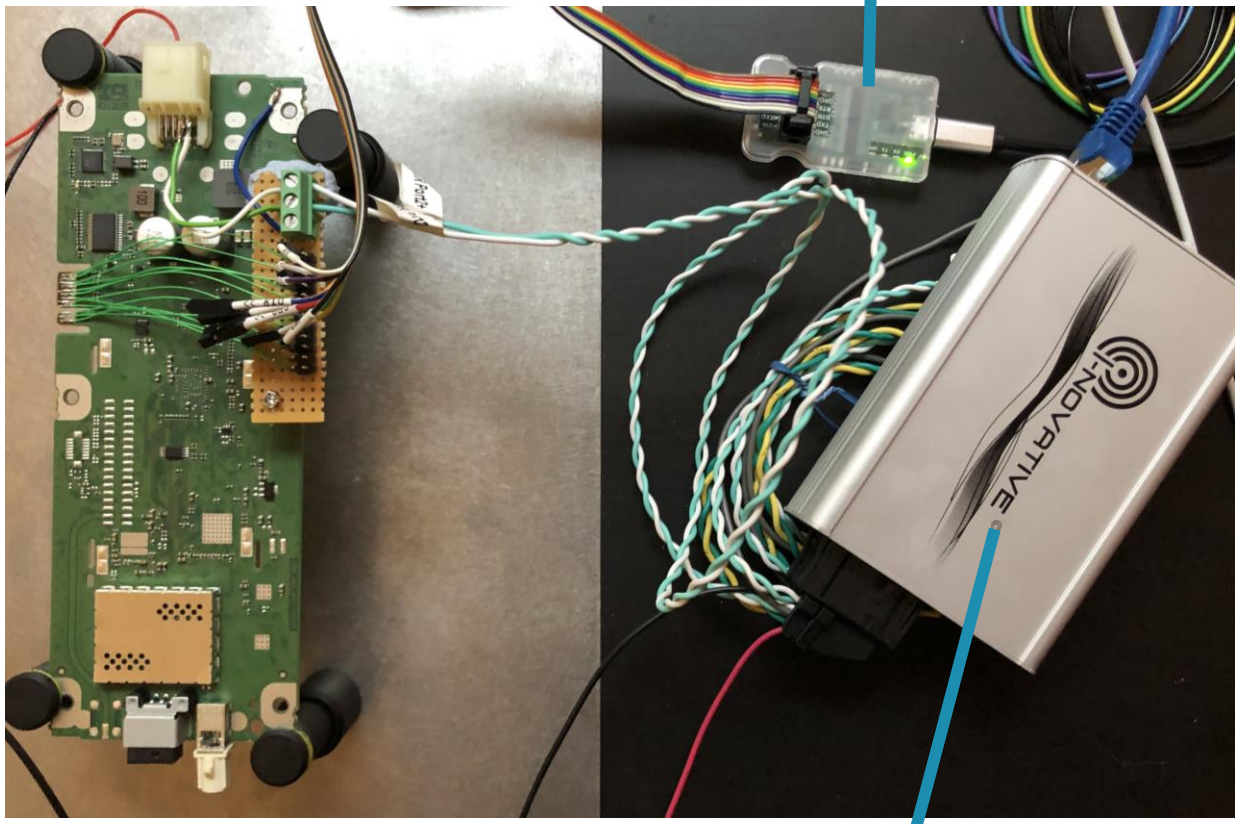



```
HARMANs Embedded Linux for Automotive 2.0.2

tunergen2-z500-a0 login: root
Password:
root@tunergen2-z500-a0:~# id
uid=0(root) gid=0(root) groups=0(root)
root@tunergen2-z500-a0:~#
```

Discovering interfaces: UART

USB to UART

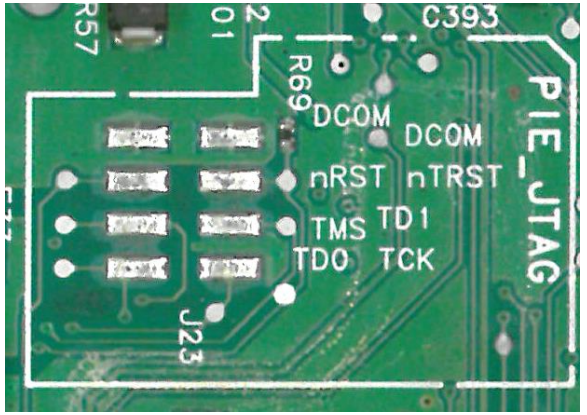


Automotive ethernet to Ethernet

Target		Prize Amount	Master of Pwn Points	Additional Prize Options
Initial Vector	*Final Stage*			
Tuner, Wi-Fi, Bluetooth, or Modem	Infotainment	$250,000	25	Infotainment Root Persistence Add-on
				CAN Bus Add-on
				Vehicle Prize
Infotainment	VCSEC, Gateway, or Autopilot	$300,000	30	Infotainment Root Persistence Add-on
				Autopilot Root Persistence Add-on
				Vehicle Prize
Tuner, Wi-Fi, Bluetooth, or Modem	VCSEC, Gateway, or Autopilot	$400,000	40	Infotainment Root Persistence Add-on
				Autopilot Root Persistence Add-on
				Vehicle Prize

https://www.zerodayinitiative.com/blog/2022/1/12/pwn2own-vancouver-2022-luanch

Discovering interfaces: JTAG



- Joint Test Action Group (IEEE Std 1149.1)
- TDI, TDO, TMS, TCK, (TRST)
- Initially designed to test chip interconnections on a PCB (Boundary Scan)
- Most manufacturers add extra functionality
  - Halt processor, step through instructions, read and write to memory

Joe Grand's JTAGULATOR
http://www.grandideastudio.com/jtagulator/)

JTAGenum
https://github.com/cyphunk/JTAGenum

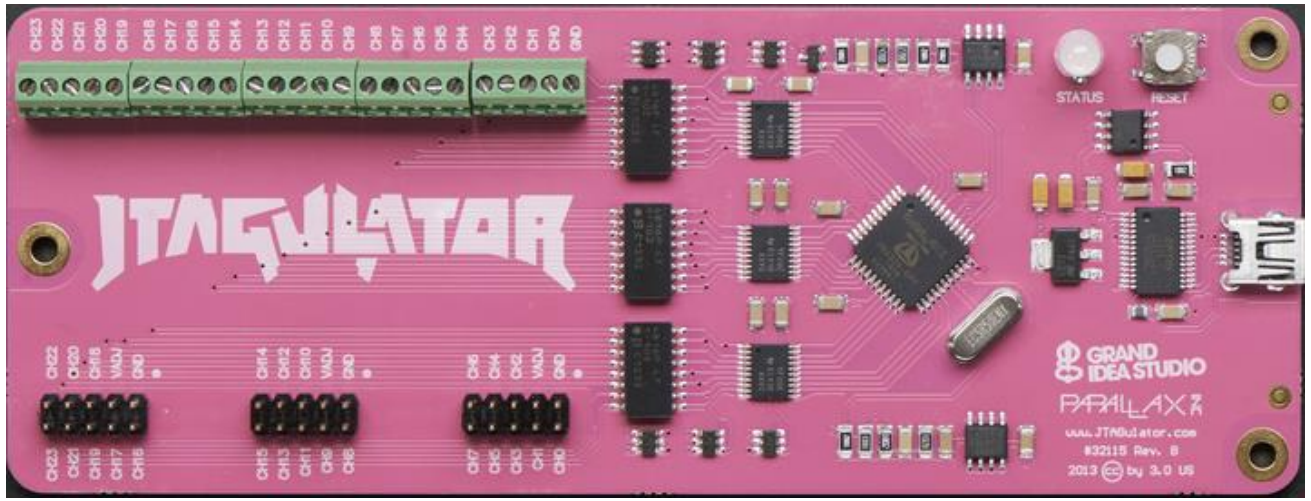


Image source: https://www.adafruit.com/product/1550

Further reading: https://blog.senr.io/blog/jtag-explained and https://wrongbaud.github.io/jtag-hdd/
(ab)using JTAG: https://github.com/syncsrc/jtagsploitation and https://vimeo.com/189122141

JTAG: controlling the TAP

- Software
  - OpenOCD (http://openocd.org/)
  - UrJTAG (http://urjtag.org/)
- Hardware
  - Segger J-Link (https://www.segger)
  - FT232H breakout

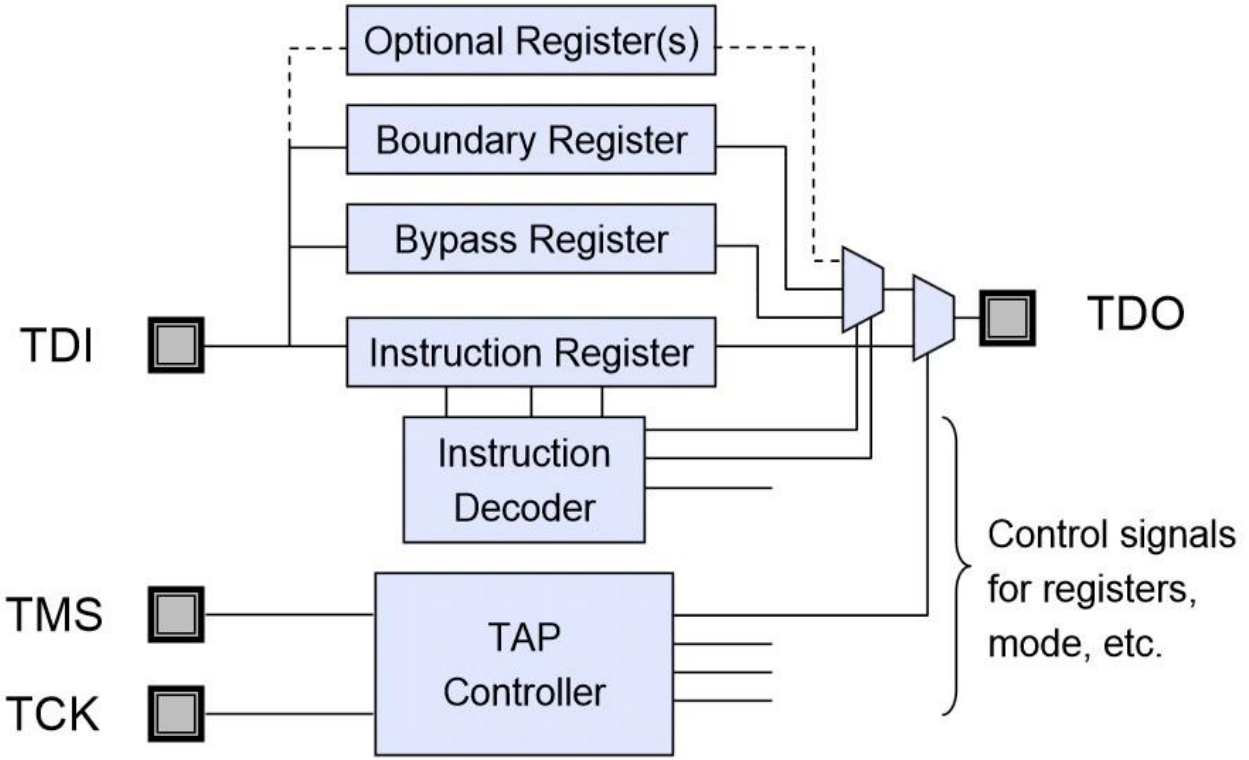
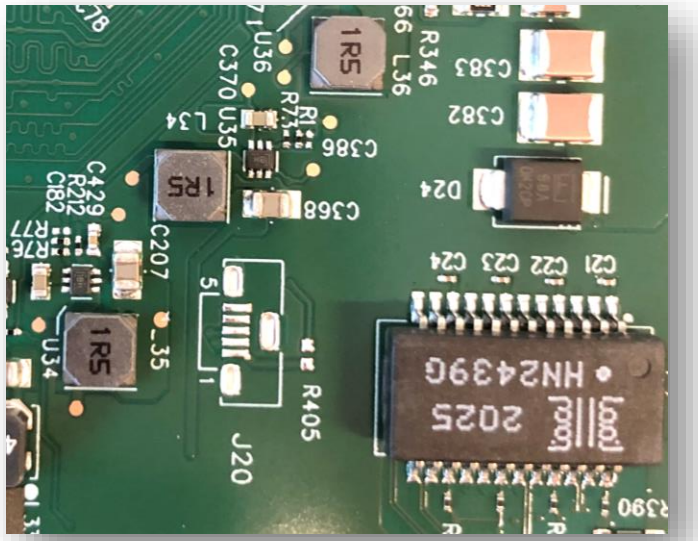
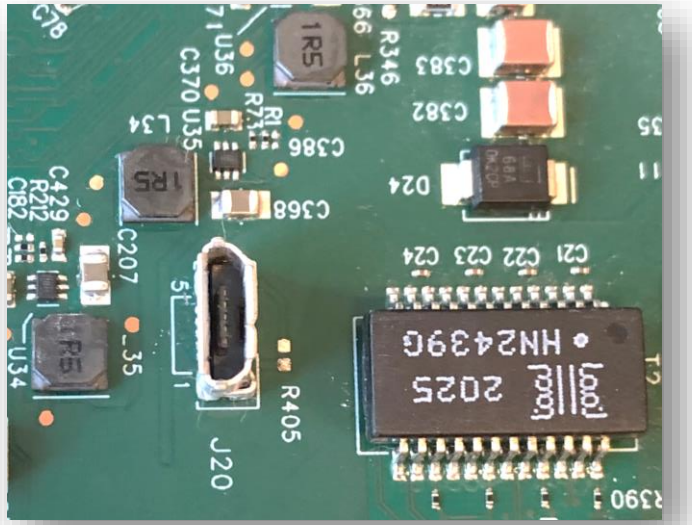


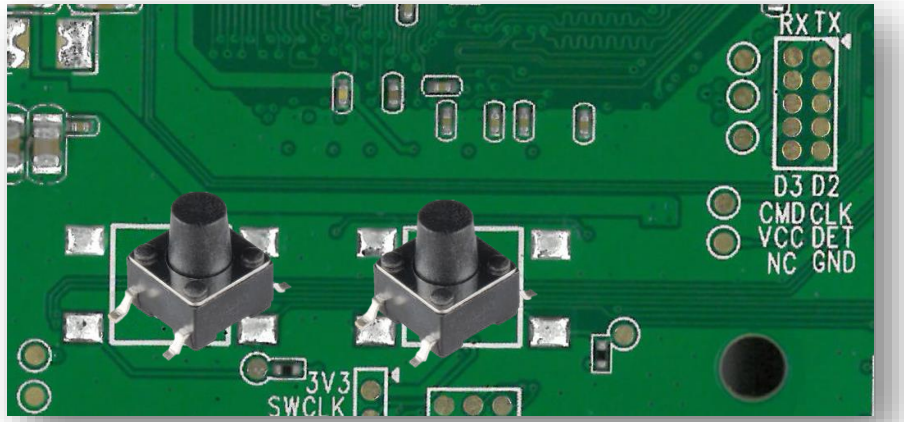
Image source: https://semiengineering.com/knowledge_centers/standards-laws/standards/ieee-1149/

PCB level modifications

- Unpopulated components and connectors
- Mode selection pull-up/pull-down resistors
- Replacing a fused component (debug interface disabled) with an unfused one
- …







COSIC

KU LEUVEN

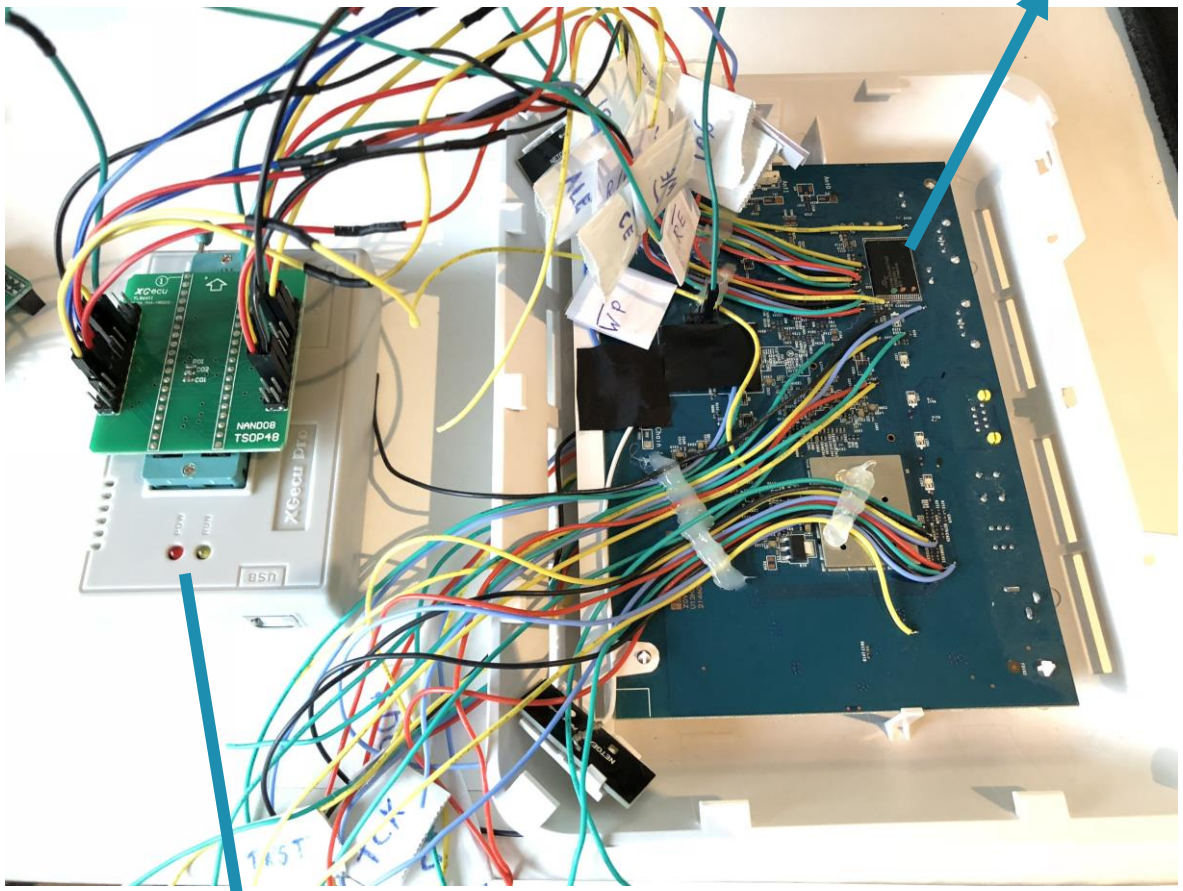
PCB level modifications

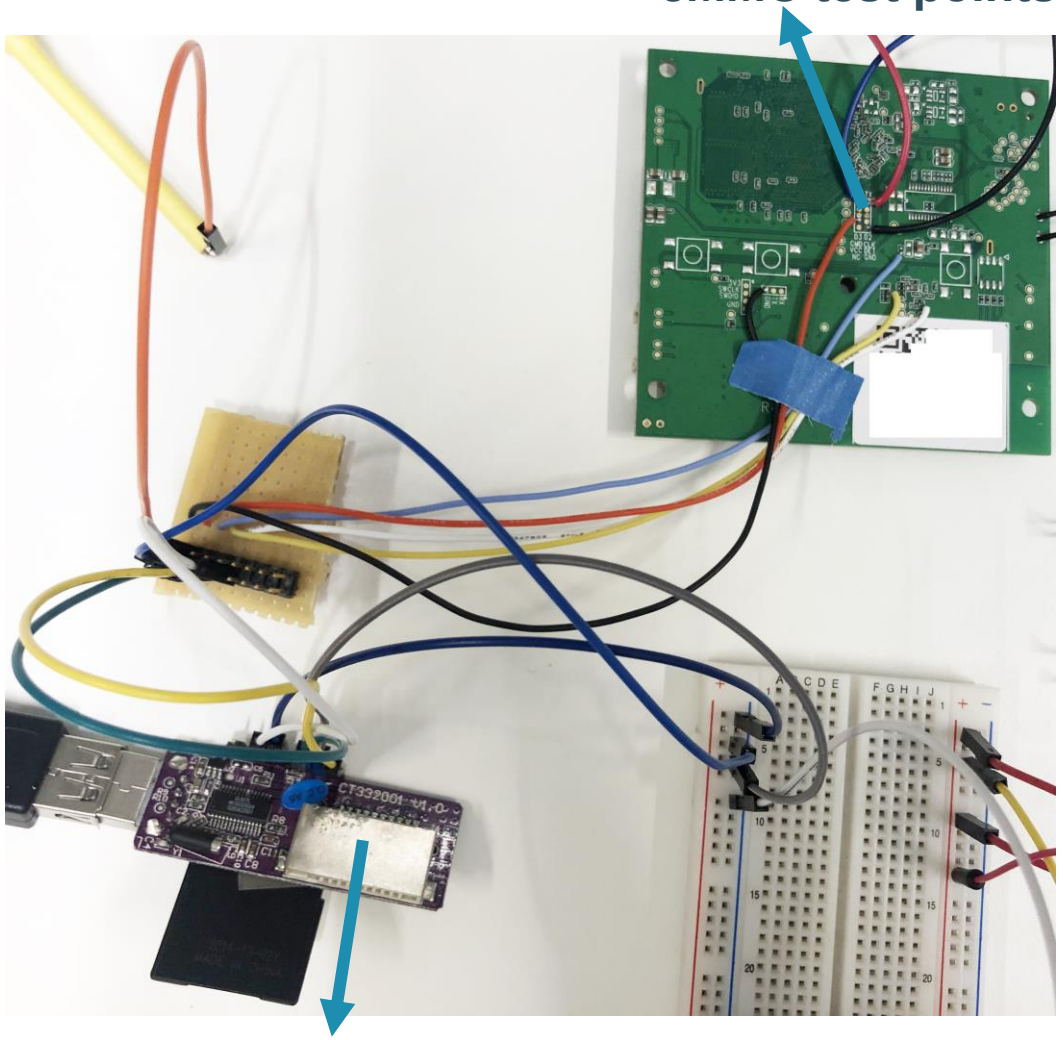
- Boot configuration pins

- Not all datasheets are public
  - Google dork:
    - filetype:pdf <partname> “confidential”
    - filetype:pdf <partname> “proprietary”

Pin #	Pin name BOOT_CONFIG[n]	Alternate function	Type	Functional description
A64	10	GPIO56	I	Mode: 0 Native mode 1 Test mode
B49	13	GPIO52	I	Apps boot source. 0 Boot from SPI NOR 1 Boot from code RAM
A90	0	GPIO3	I	Apps authentication enable. Enables authentication for various AP code segments. Send to security control. 0 No authentication 1 Enable authentication
A63	9	GPIO55	I	Force USB boot. 0 Normal boot 1 Force boot ROM USB interface. Send to security control.
A68	11	GPIO62	I	JTAG boot enable. 0 GPIO0~GPIO5 are normal GPIOs; can be configured by the FUNC_SEL registers. 1 GPIO0~GPIO5 are occupied by JTAG interface; cannot be changed by the FUNC_SEL registers.
B55	14	GPIO61	I	Watchdog disable. Only valid in native mode. 0 Watchdog enabled 1 Watchdog disabled

Reading non-volatile memory: in-circuit

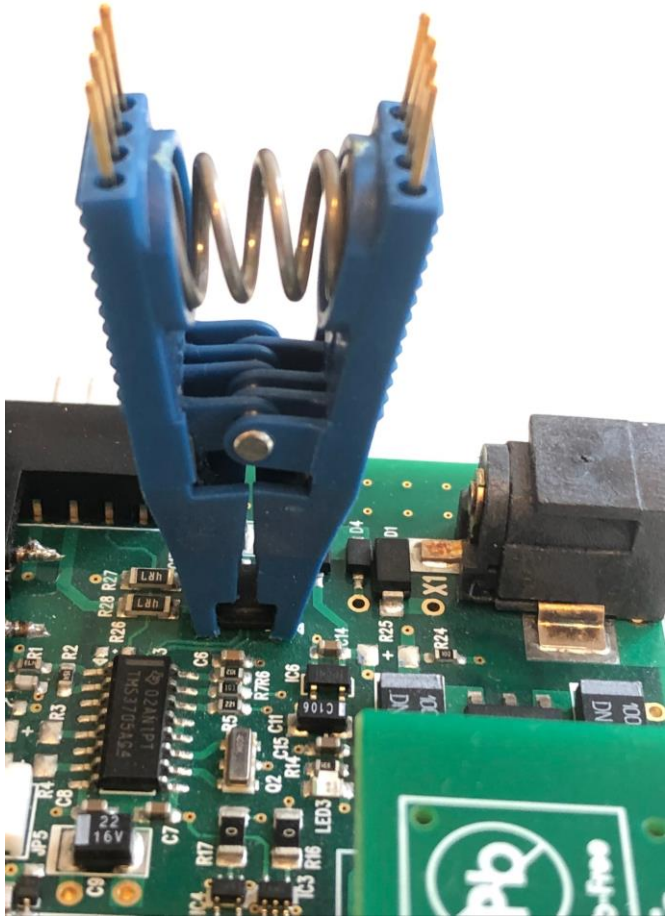




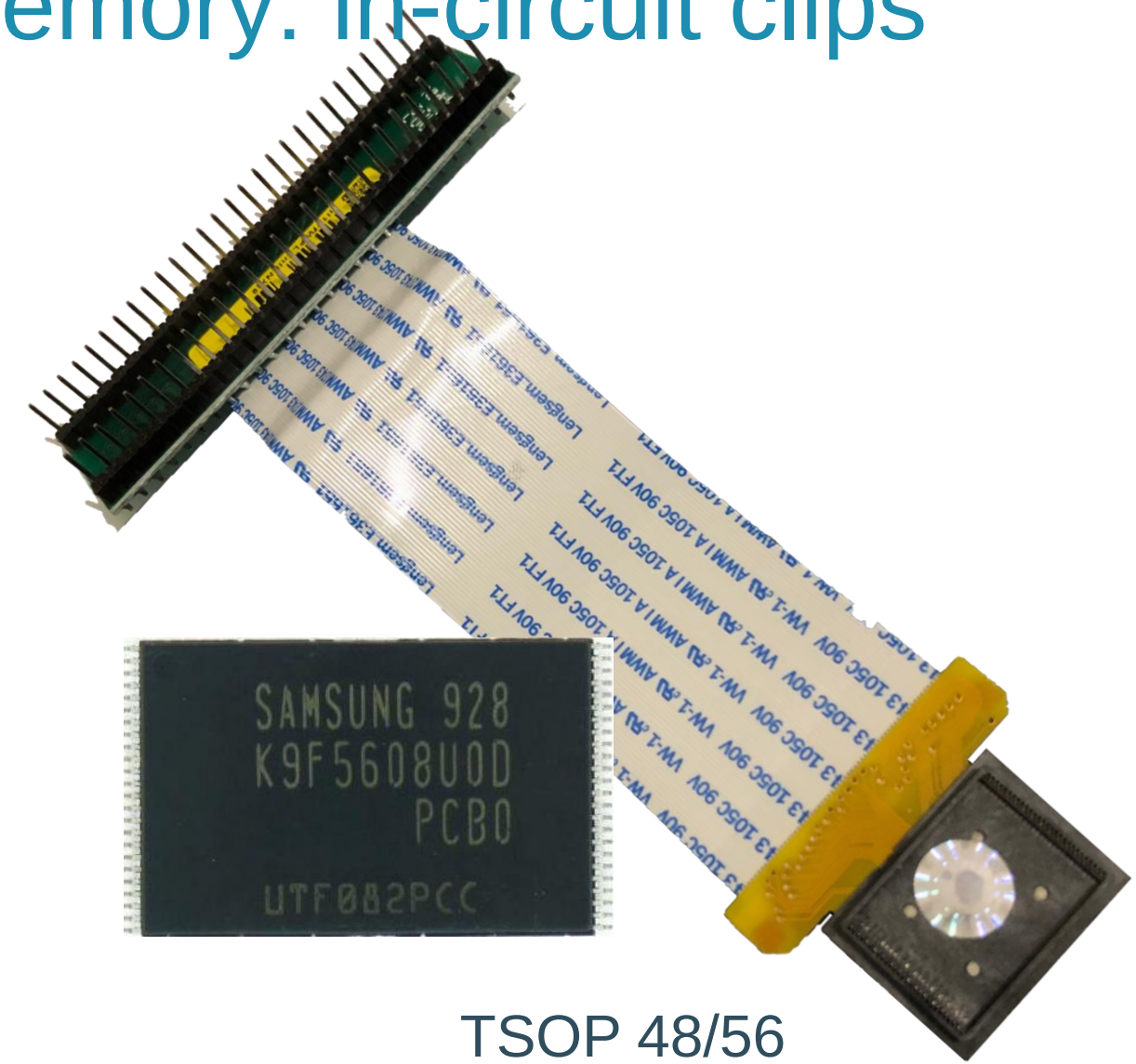
COSIC

KU LEUVEN

Reading non-volatile memory: in-circuit clips



SOIC 8/14/16/20/24/28/32/40



TSOP 48/56

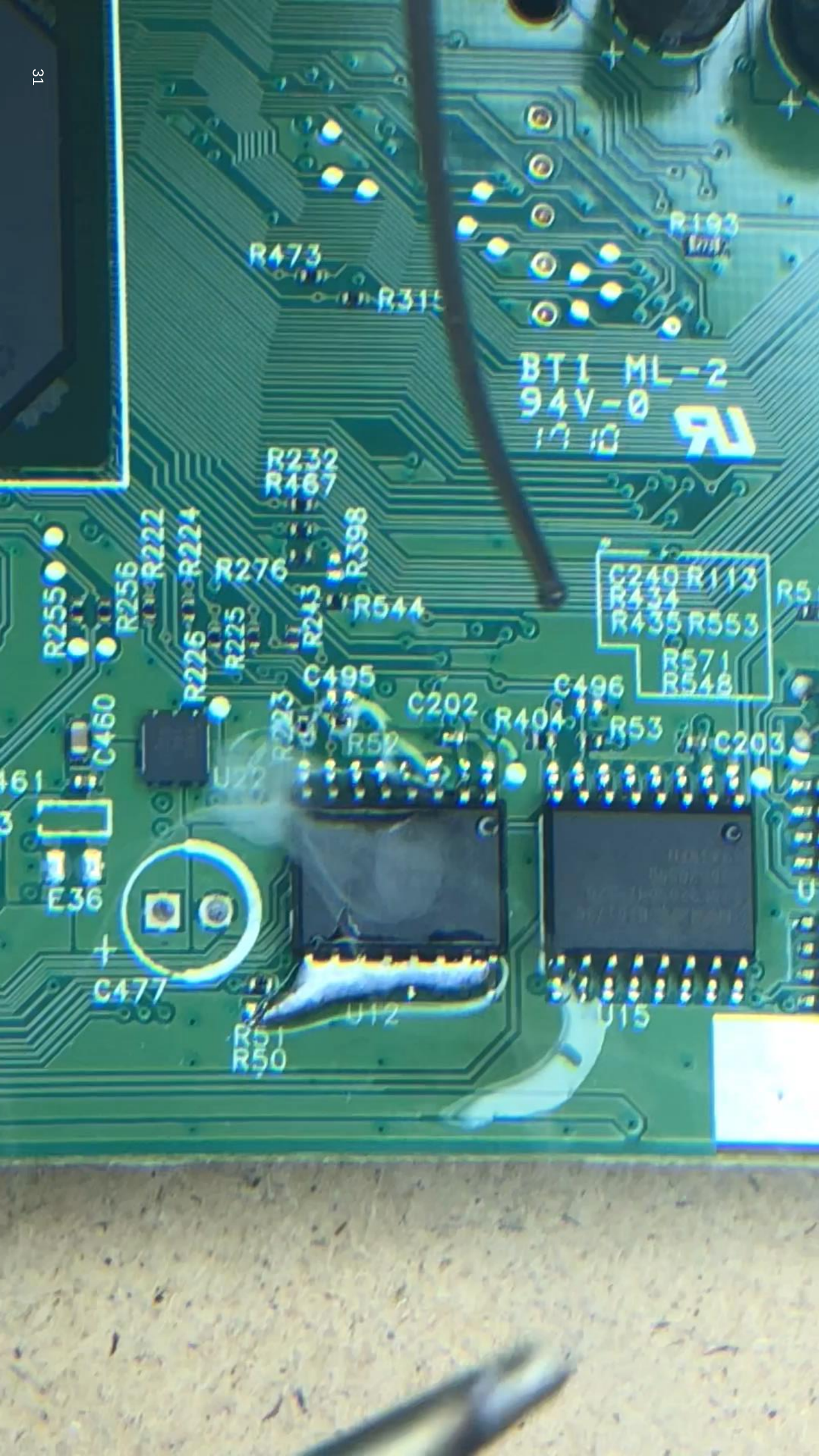
Dumping non-volatile memory: chip-off

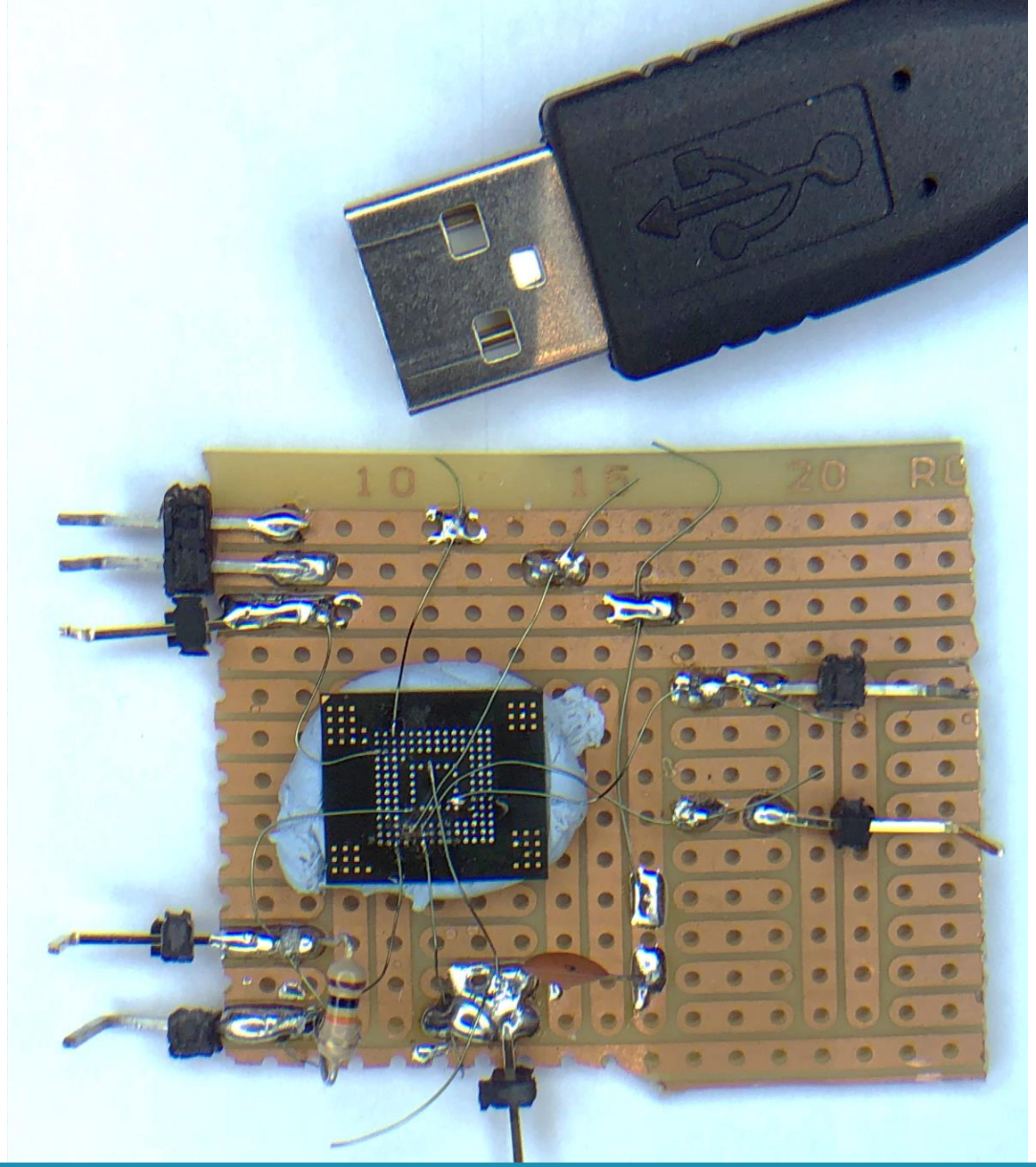
- Hot air rework station
- ChipQuick SMD1 alloy (https://www.chipquik.com/)
- (CNC) Mill

Source: atten.eu

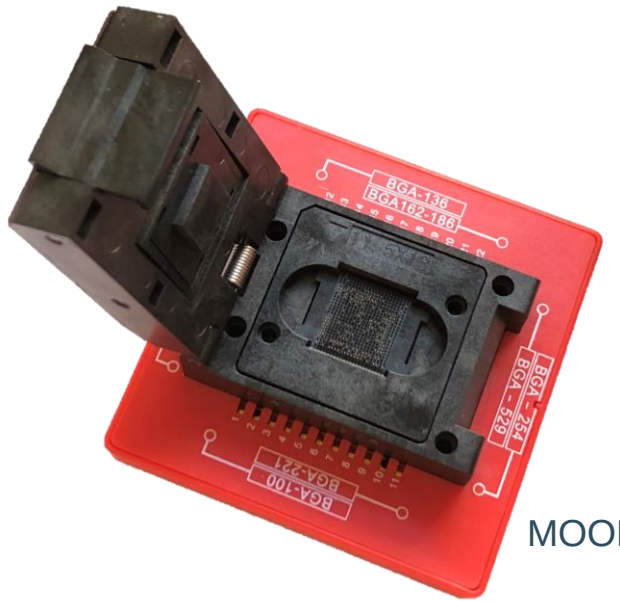
Source: ultratecusa.com/product/blue-mill

Step by step chip-off using hot air: https://forensicswiki.xyz/wiki/index.php?title=Chip-Off_Forensics


R473
R315
R193
BTI ML-2
94V-0
R232
R467
R276
R398
R222
R224
R256
R255
R544
R225
R226
C495
C202
R404
R53
C203
C460
C240
R113
R434
R435
R553
R571
R548
E36
C477
U12
U15
R51
R50



source: aliexpress.com



MOORC eMMC socket

COSIC

KU LEUVEN

Reading non-volatile memory: readers/sockets

elnec.com

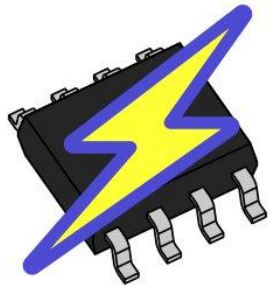
dataman.com

xgecu.com

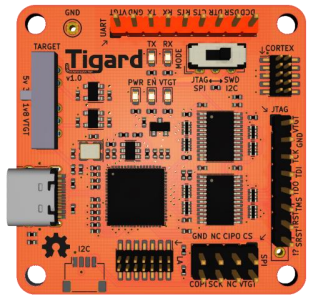
batronix.com

easy-jtag.com

allsocket.com



flashrom.org



github.com/tigard-tools

raspberrypi.org

COSIC

KU LEUVEN

Dissecting the memory dump

- Raw NAND flash dumps may need preprocessing
  - https://www.j-michel.org/blog/2014/05/27/from-nand-chip-to-files

- Binwalk! (https://github.com/ReFirmLabs/binwalk)
  - Will try to identify common file types in a binary file
  - Entropy graphs help determine compressed/encrypted regions
  - In many cases allows to easily extract a linux filesystem

Further reading: https://www.pentestpartners.com/security-blog/how-to-do-firmware-analysis-tools-tips-and-tricks/

What if

- Debug interfaces do not accept input
- Debug interfaces not accessible or disabled
- No easily exploited software vulnerability
- External (non-)volatile memory is encrypted
- The device implements verified/secure boot

Physical attacks

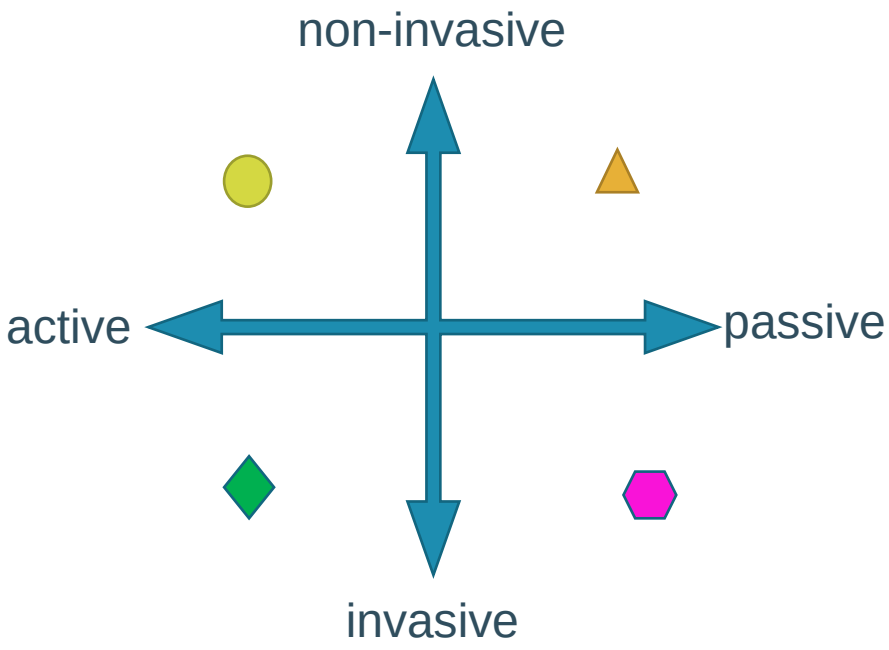
Physical attacks (classical model)

- Passive versus active
  - Passive: observe, device operates normally
  - Active: perturbate
- Invasiveness
  - Non-invasive: closed package
  - Invasive: open package, contact chip
  - Semi-invasive: open package, no contact

} Sample prep

- Many techniques are derived from reliability and failure analysis!

- Power side-channel attack
- Voltage fault injection
- Circuit modification
- Microprobing

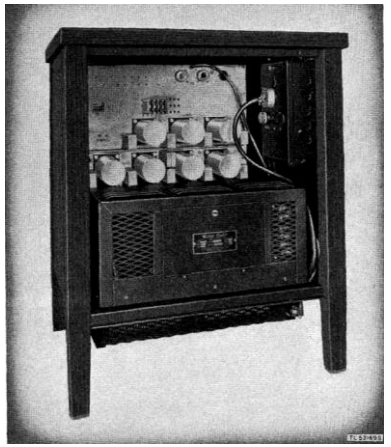


Side-channel analysis

- Measure and record a physical observable (a side-channel)
  - Often combine multiple recordings or traces
- Analyze these traces to reveal secret information
  - Basic observation, statistical techniques, machine learning

nytimes.com/2018/04/19/nyregion/master-manipulator.html

wikipedia.org/wiki/German_tank_problem



wikipedia.org/wiki/Tempest_(codename)
wikipedia.org/wiki/Van_Eck_phreaking

Differential Power Analysis – Kocher, Jaffe and Jun - https://paulkocher.com/doc/DifferentialPowerAnalysis.pdf

Listening to chips

AKA Side-Channel Analysis

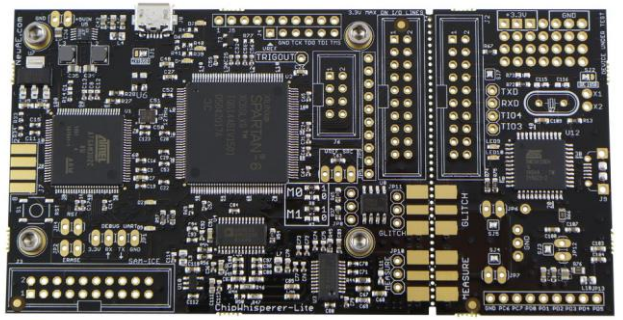
- Power consumption
- EM emanations
- Timing
- Photon emissions
- Temperature
- Sound
- …

Allied Vision

(semi-invasive)

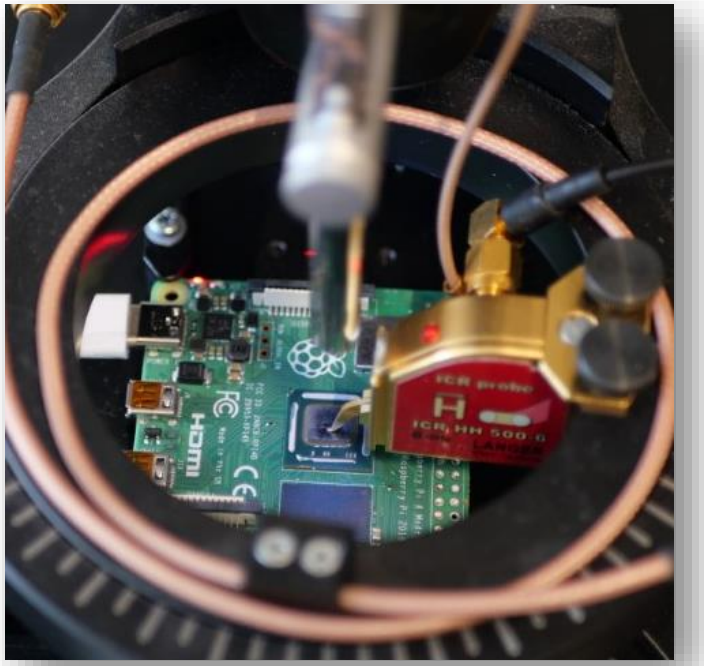
Short-wave infrared camera for emission microscopy

Pico Technology - oscilloscope



ChipWhisperer by NewAE

Langer Localised EM



Langer Near-Field Probes

ChipWhisperer: https://github.com/newaetech/chipwhisperer

COSIC

KU LEUVEN

Fault injection

- Intentionally inject a (transient) fault in an integrated circuit
  - (single-even upset is unintentional)
- Violates the classic model we have of a processor
  - It may no longer execute the instructions you programmed
- Many different FI techniques exist

Wildthing Unlooper
https://dsssupplies.tripod.com/wildthing.htm

- First (?) used for Pay-TV piracy
  - Card were forced into infinite loop
  - Unlooper would glitch a card past that loop

https://twitter.com/akacastor/status/1054964836923334656

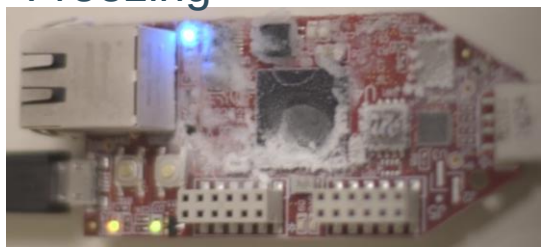
Thread on DirecTV hacking by Chris Gerlinksy: https://threadreaderapp.com/thread/1054964796506939392.html
Design Principles for Tamper-Resistant Smartcard Processors – Kömmerlink and Kuhn - https://www.cl.cam.ac.uk/~mgk25/sc99-tamper.pdf

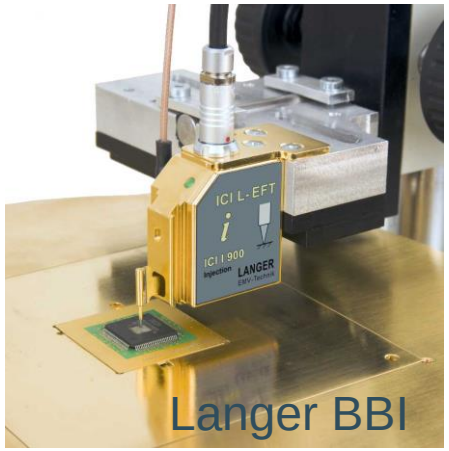
Torturing chips

AKA Fault Injection

- Tools of the trade:
  - Voltage Fault Injection
  - Clock glitching
  - EM Fault Injection
  - Laser Fault Injection
  - Body Biasing Injection
  - Extreme temperatures
  - UV light
  - Radiation
  - …

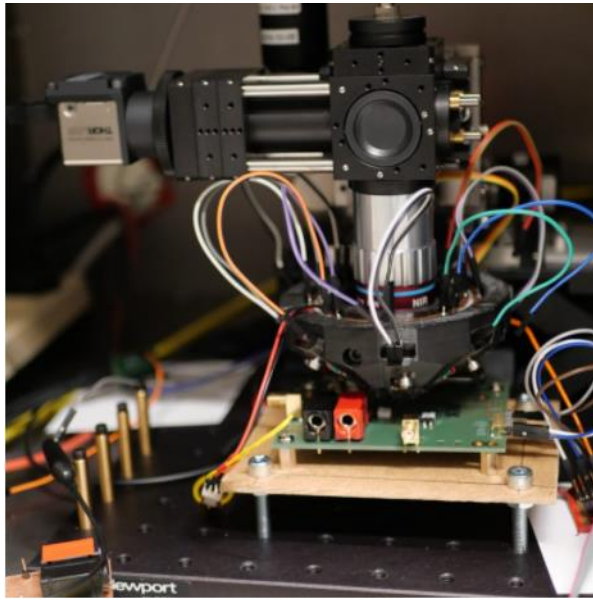
Freezing

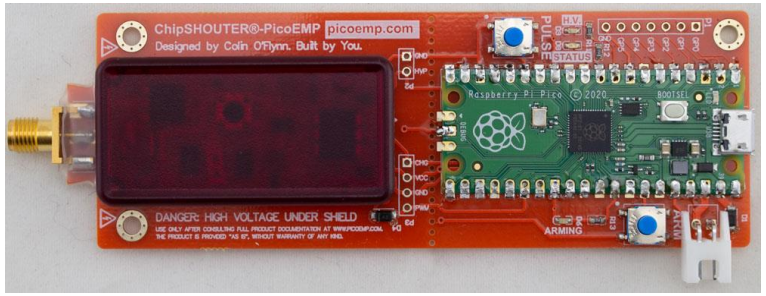




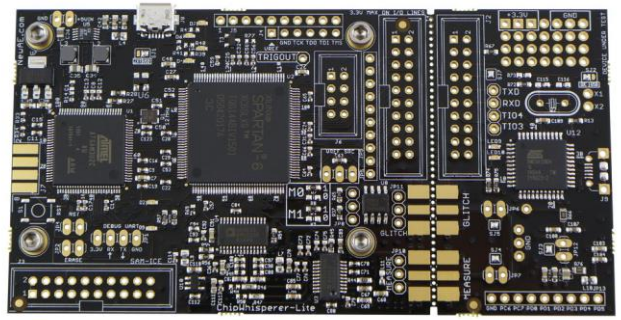
Langer BBI

Laser FI





https://github.com/newaetech/chipshouter-picoemp



ChipWhisperer and ChipShouter by NewAE

ChipWhisperer: https://github.com/newaetech/chipwhisperer
PhD thesis S, Skorobogatov: https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-630.html
Bypassing CRP overview: https://www.youtube.com/watch?v=DTuzuaiQL_Q

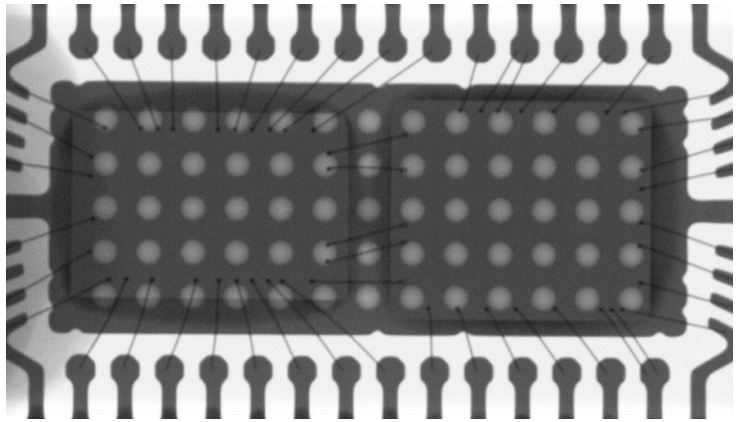
COSIC
KU LEUVEN

Imaging

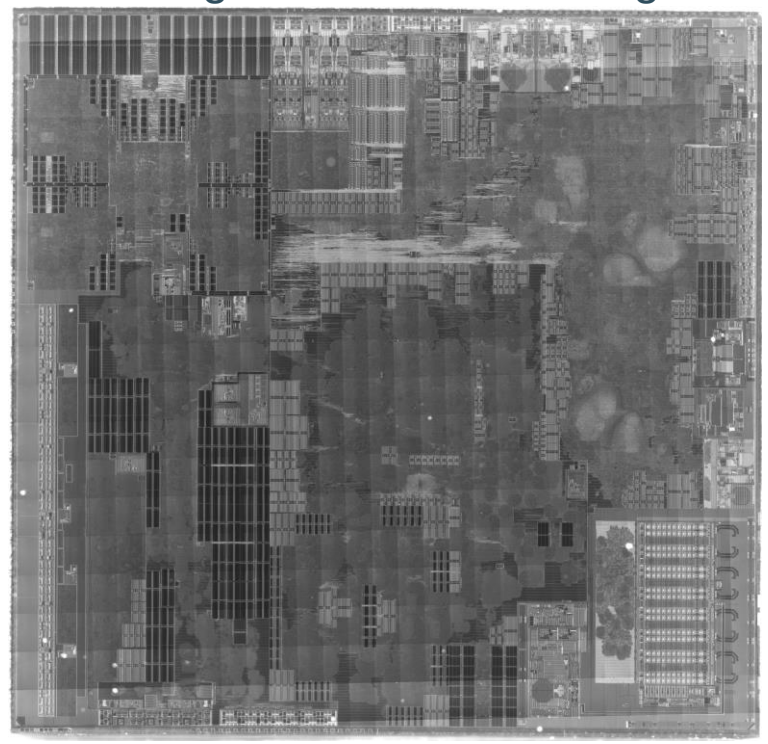
- Many different techniques exist
  - Optical microscopy
  - Scanning Electron Microscope (SEM)
  - (Ptychographic) X-Ray

- Optical ROM extraction
  - Capturing Mask ROMs – John McMaster – Hardwear.io 2020
    - https://www.youtube.com/watch?v=JpA2VScMqHo
- Circuit reverse engineering
  - Reverse-engineering a cryptographic RFID tag - Nohl et al. – USENIX 2008
    - https://www.cs.virginia.edu/~evans/pubs/usenix08/usenix08.pdf

X-Ray



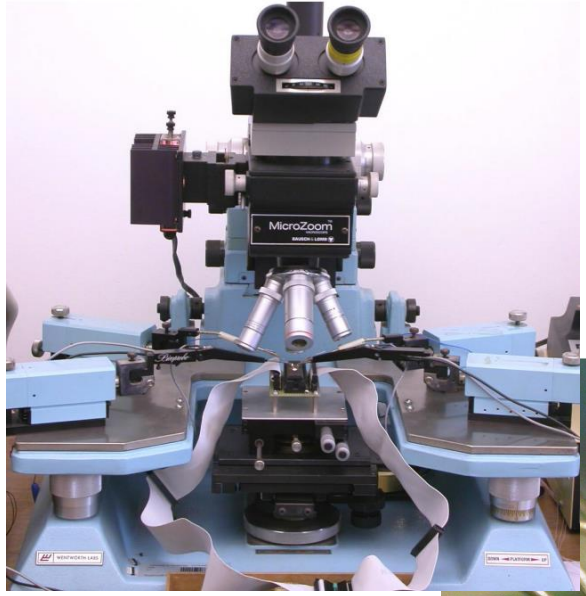
Through substrate die image

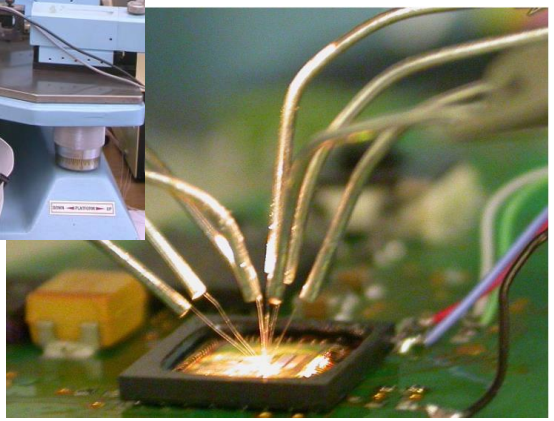


Collection of die images: https://siliconpr0n.org/map/

Probing

- Microprobing station
  - Accurate positioning of very small needles





- Can be used to sniff chip internal signals (data bus)
  - How microprobing can attack encrypted memory – Sergei Skorobogatov
    - https://www.cl.cam.ac.uk/~sps32/ahsa2017_prob.pdf
- Externally bridge contacts with multiple probes (fuses)
  - Extracting programmable logic with photons - John McMaster – Hardwear.io 2021
    - https://www.youtube.com/watch?v=pUFftFKe3YI

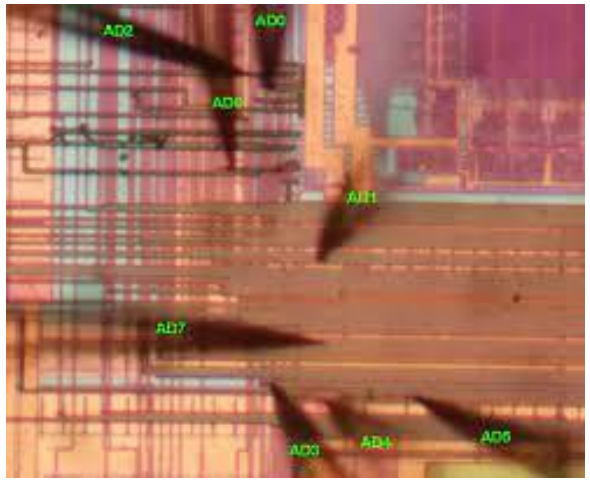
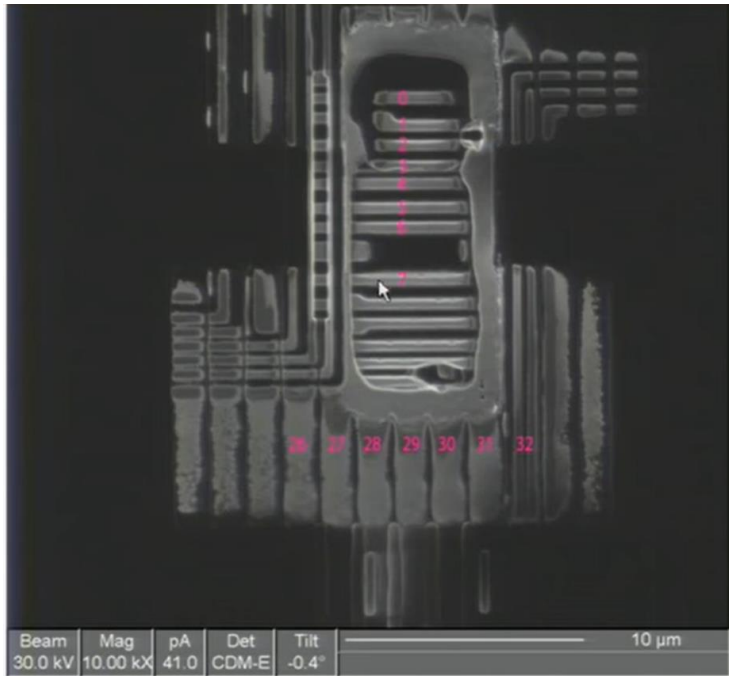


Image source: https://www.cl.cam.ac.uk/~sps32/ahsa2017_prob.pdf

COSIC KU LEUVEN

Circuit modification

- Focused Ion Beam (FIB)
- Can deposit and remove parts of an IC
- Requires skilled operator



Christopher Tarnovsky – Deconstructing a ‘secure’ processor
https://www.youtube.com/watch?v=62DGIUpscnY

en.wikipedia.org/wiki/Focused_ion_beam

Case studies

COSIC

KU LEUVEN

The Hackable Home

- How secure are “smart” devices?
- What is the privacy impact?

TEST aankoop

euroconsumers

COSIC

vrt NWS

Pas op voor goedkope slimme apparaten: "Ze zijn zo lek als een zeef"

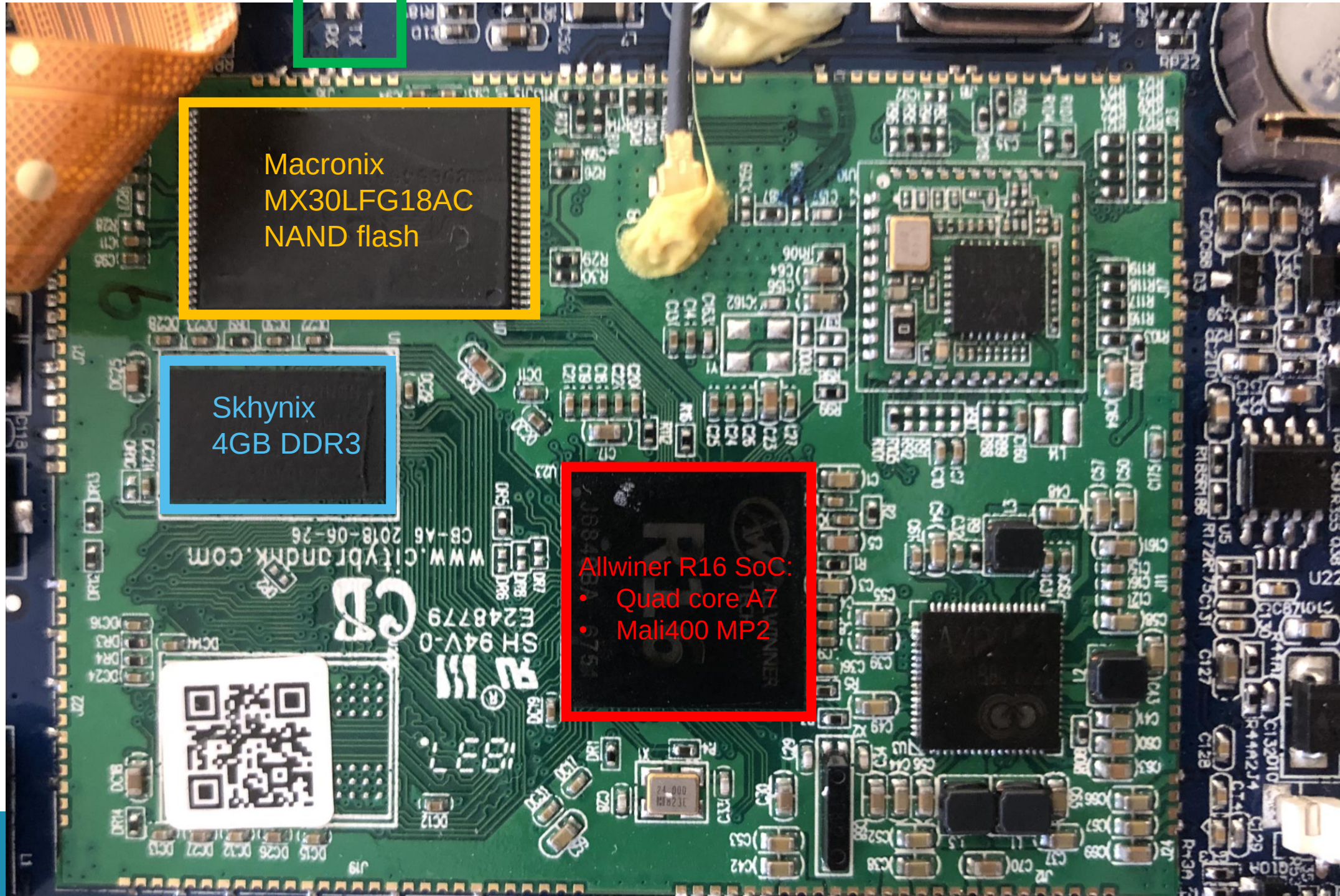
Slimme huishoudtoestellen zoals een deurbel, stofzuiger of wasmachine zijn heel makkelijk te hacken. Dat blijkt uit een onderzoek van Europese consumentenorganisaties, waaronder Test Aankoop, en de KU Leuven. Er werden maar liefst 54 kwetsbaarheden ontdekt bij de 16 onderzochte producten. De onderzoekers konden makkelijk wachtwoorden onderscheppen en een slim deurslot kon vanop afstand worden geopend.

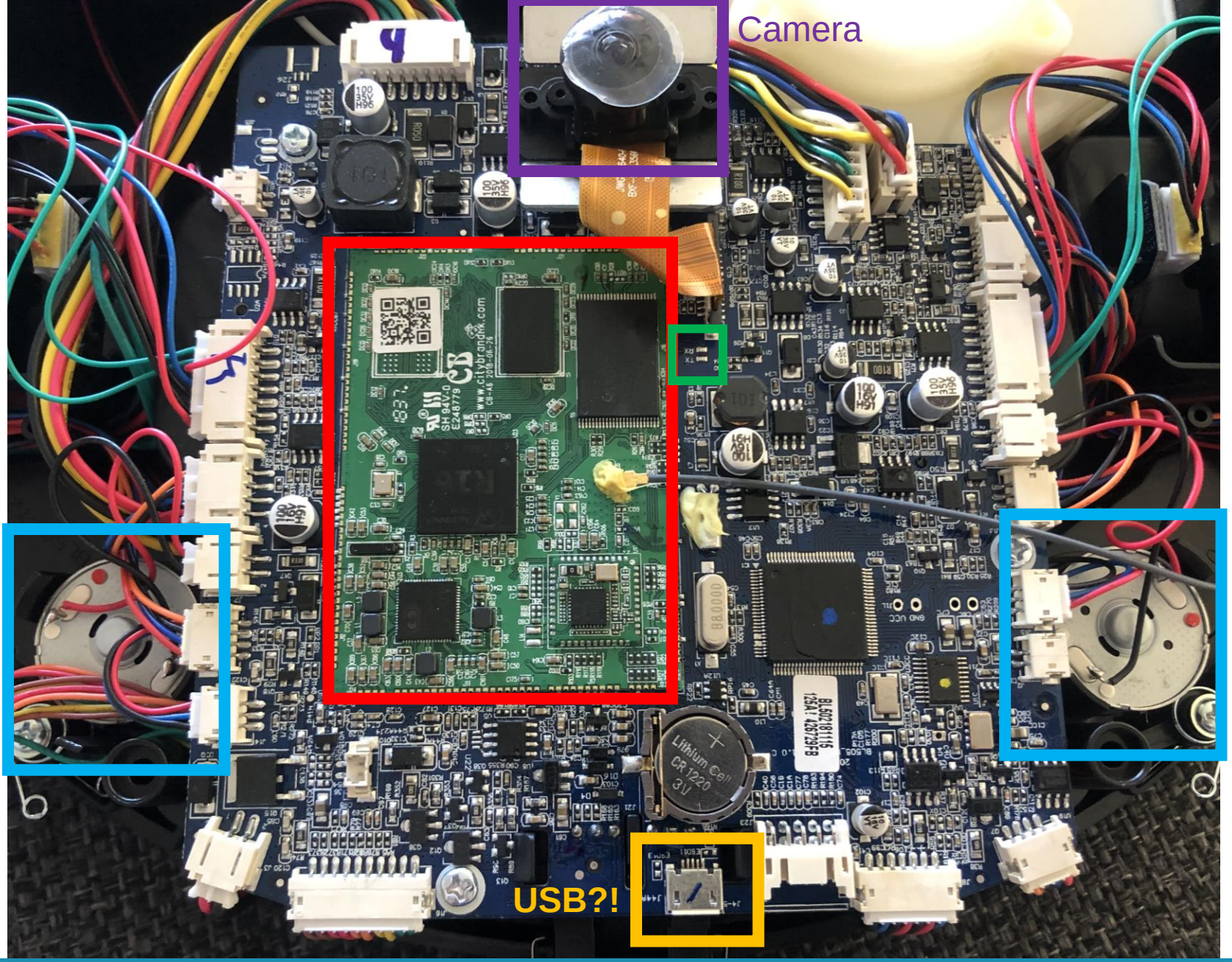
Heel handig, maar gevaarlijk

Een slimme deurbel, wasmachine of stofzuiger zijn tegenwoordig niet meer weg te denken uit sommige huishoudens. Omdat ze verbonden zijn met het internet zijn ze heel handig in gebruik. Zo kan je vanop je smartphone zien wie aan de deur belt, of kan je de living laten stofzuigen terwijl je op het werk zit.

https://www.vrt.be/vrtnws/nl/2021/08/25/pas-op-voor-goedkope-slimme-toestellen-ze-zijn-zo-lek-als-een/

UART Debug connector





COSIC

KU LEUVEN

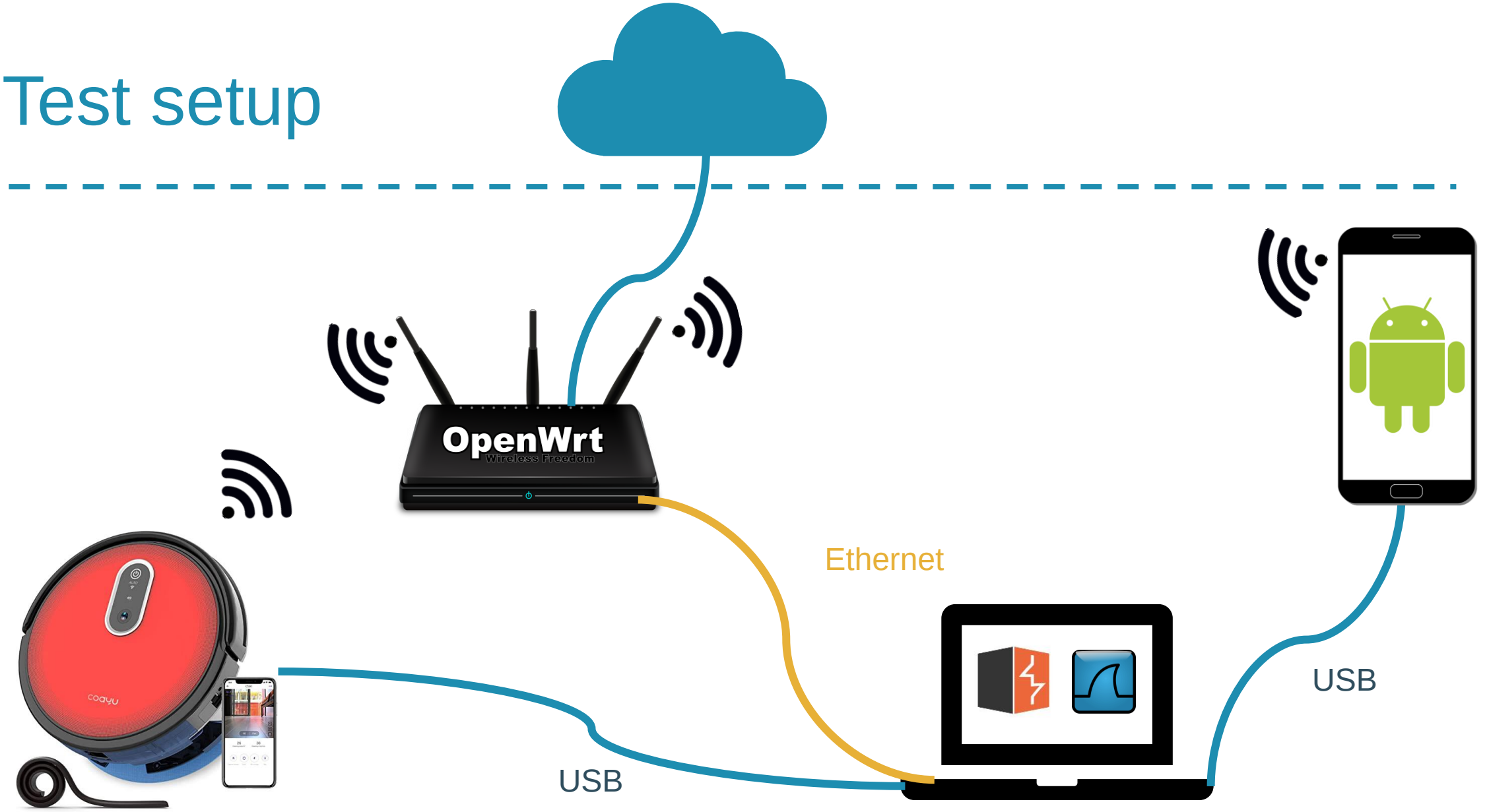
Android Debug Bridge shell over USB

```
BusyBox v1.27.2 () built-in shell (ash)


 _____  _                    __      _
|_   _||_| ___  _ _    |  |    |_| ___  _ _  _ _
  | |    _ |   ||   |  |  |__  | ||   || | ||_'_|
  | |  | || | || _ |  |_____||_||_|_||___||_,_|
  |_|  |_||_|_||_|_|  Tina is Based on OpenWrt!
 ----------------------------------------------
 Tina Linux (Neptune, 5A0C0971)
 ----------------------------------------------
root@TinaLinux:/# id
uid=0(root) gid=0(root)
root@TinaLinux:/#
```

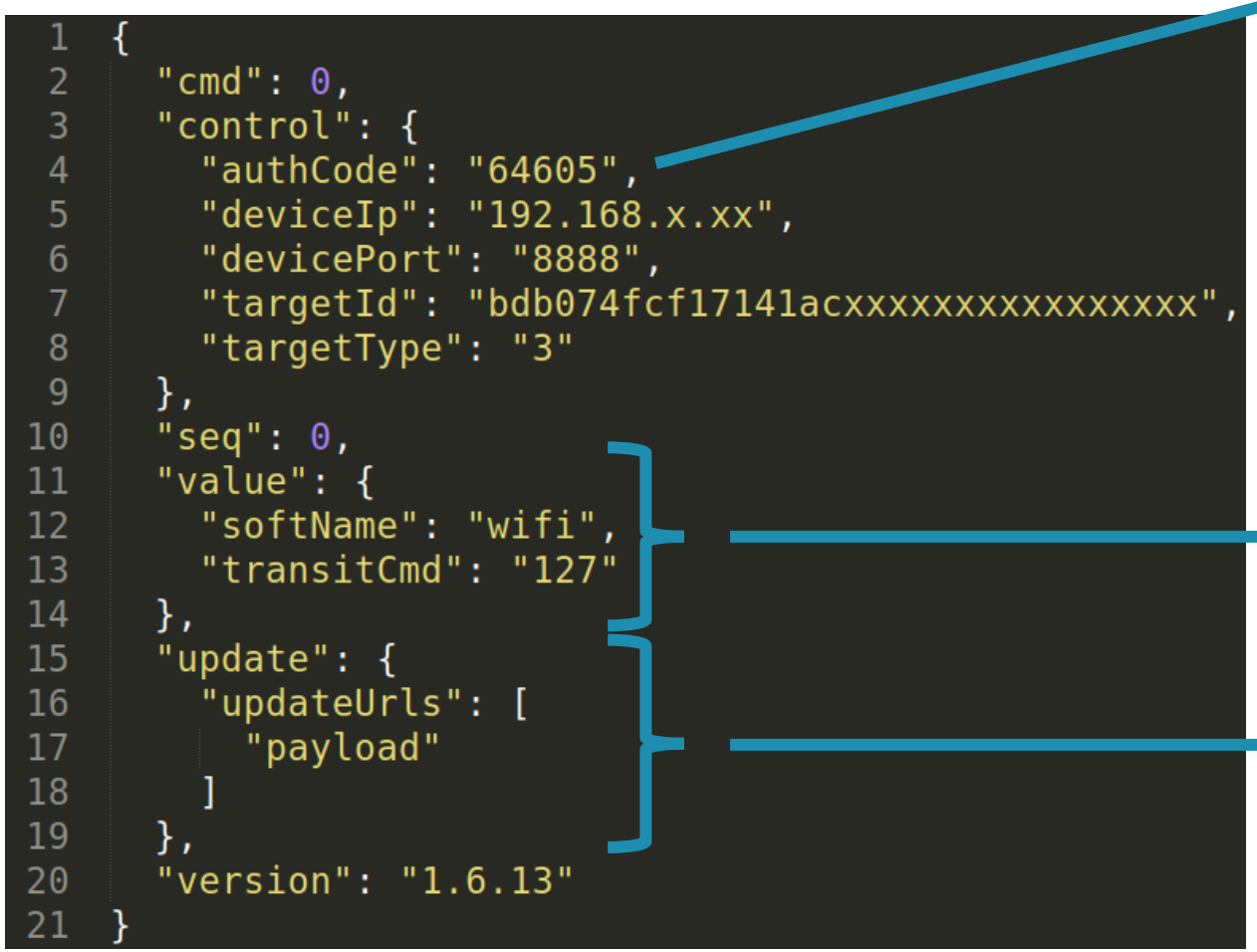

More info on ADB: https://developer.android.com/studio/command-line/adb

Test setup



Android APP reversing: https://ragingrock.com/AndroidAppRE/ and youtube.com/watch?v=BijZmutY0CQ
MiTM setup: https://github.com/koenbuyens/kalirouter and https://www.mickaelwalter.fr/use-openwrt-as-a-mitm-router/

Findings (1)



```
{
  "cmd": 0,
  "control": {
    "authCode": "64605",
    "deviceIp": "192.168.x.xx",
    "devicePort": "8888",
    "targetId": "bdb074fcf17141acxxxxxxxxxxxxxxxx",
    "targetType": "3"
  },
  "seq": 0,
  "value": {
    "softName": "wifi",
    "transitCmd": "127"
  },
  "update": {
    "updateUrls": [
      "payload"
    ]
  },
  "version": "1.6.13"
}
```

- Robot will only accept commands if the authCode is correct
- Can be obtained from the API given a valid targetID

- Robot ID
- API allows enumeration

- Perform command 127
- With the softName argument

- The update can be downloaded from this URL

Findings (2)

Ghidra decompiler output

Strings help to find interesting bits of code!

```
62  if (DAT_0002a3d8 == 1) {
63    puts("ota is progressing now!!!");
64  }
```

…

Kills a process

Removes a file

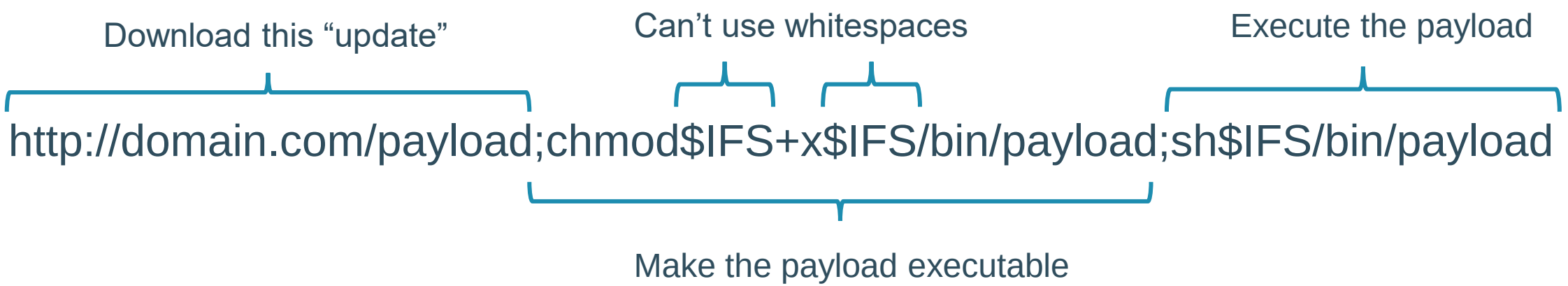
Downloads a file…

```
102       else {
103         sprintf(acStack956,"killall -9 %s",&DAT_0002a3dc);
104         printf("cmd_kill_bin is %s\n",acStack956);
105         system(acStack956);
106         sprintf(acStack892,"rm %s -rf",&DAT_0002a3dc);
107         printf("cmd_rm_bin is %s\n",acStack892);
108         system(acStack892);
109         sprintf(acStack828,"/bin/wget %s -P /bin 2>/tmp/ota.txt &",&UPDATE_URL);
110         printf("cmd_wget_bin is %s\n",acStack828);
111         system(acStack828);
112       }
113     }
```


Ghidra: https://ghidra-sre.org/
No Starch Ghidra book: https://nostarch.com/GhidraBook

Findings (3)

- The robot parses packets received from the backend
  - One specific packet triggers a software update and includes a URL
  - The update is downloaded using wget, the URL is not sanitized
  - We can perform OS command injection through the URL parameter!



OS Command Injection: https://portswigger.net/web-security/os-command-injection
Bypassing filters: https://github.com/swisskyrepo/PayloadsAllTheThings

Creating a robot vacuum botnet

1. Enumerate Robot IDs using an API flaw
2. Second API flaw allows to obtain the “authCode”
3. Using the Robot ID and authCode send a malicious update command
   - The malicious command exploits a command injection vulnerability
   - We can execute any code, on any robot, without any user interaction

How big is my (hypothetical) robot network?

- The robot is a white label product
  - You buy a robot with customized packaging and APP look
- At least 25 brands
- Production capacity of 50k robots per month
- >100k APP installs on the Google Play store

- It is trivial to start your own robot brand with your own “customized” firmware
  - These robots have a microphone, speaker and camera

Motorola Comfort connect 85

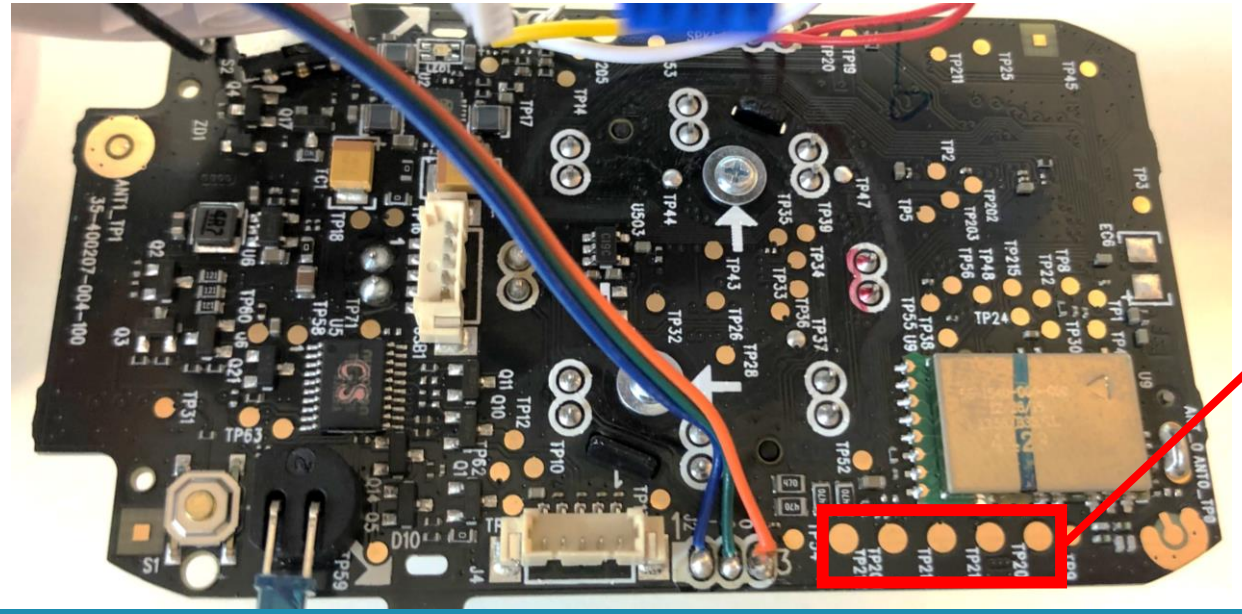
- Relatively expensive baby monitor (€250)
- Sold under the Motorola brand
  - Not actually made by Motorola…
- Camera connects to WiFi
  - Can be controlled from an APP

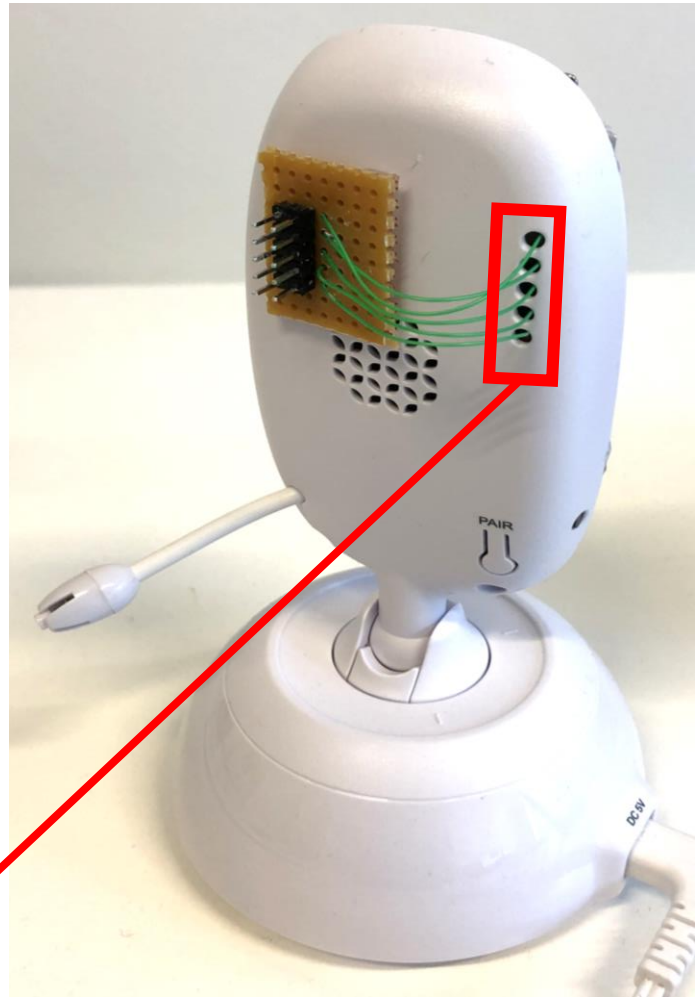
COSIC

KU LEUVEN

Motorola: hardware

- UART debug interface externally accessible
  - No immediate root shell
  - Access to U-Boot allows to dump firmware
- Firmware dump helps to analyze the system





COSIC

KU LEUVEN

Motorola: Firmware updates

- APP fetches download URL
  - https://ota.hubble.in/ota1/0335_patch/0335-03.30.42.fw.pkg
- Firmware updates are encrypted (and signed?)
- Reverse engineered the decryption process
  - Encrypted using AES-CBC
  - IV is fixed and hardcoded in the update binary
  - Encryption key is derived from the filename
  - CBC reset after 1024 bytes (64 AES blocks)
  - Last block (less than 1024 bytes) is not encrypted

Decompile: FUN_000088b4 - (aes_decrypt_full)

```
88      else {
89        memcpy(&local_42c,(void *)(argv[1] + local_14),4);
90        memcpy(&local_438,(void *)(argv[1] + local_14 + 5),8);
91        snprintf((char *)&key,0x11,"%4s%8sHUBL",&local_42c,&local_438);
92        load_key_and_iv(&key,"HubbleCvisionVN");
93        outputfile = fopen(acStack1356,"wb+");
94        if (outputfile == (FILE *)0x0) {
95          fprintf(stderr,"ERROR: Opening output file (%s) failed. Abort!\n",acStack1356);
96          fclose(inputfile);
97          uVar2 = 0xfffffffe;
98        }
99        else {
100         do {
101           local_1c = fread(auStack1060,1,0x400,inputfile);
102           if ((int)local_1c < 1) {
103 LAB_00008f3c:
104             fclose(inputfile);
105             fclose(outputfile);
106             return 0;
107           }
108           if ((int)local_1c < 0x400) {
109             local_18 = fwrite(auStack1060,1,local_1c,outputfile);
110             if ((int)local_18 < (int)local_1c) {
111               printf("Error Writing - in %d - out %d\n",local_1c,local_18);
112               fclose(inputfile);
113               fclose(outputfile);
114               return 0xffffffff;
115             }
116             goto LAB_00008f3c;
117           }
118           local_18 = do_decrypt(auStack1060,local_1c);
119           if (local_18 != local_1c) {
120             if (local_c == 0) {
121               pcVar3 = "Decrypt";
122             }
123             else {
124               pcVar3 = "Encrypt";
125             }
126             printf("Error %s\n",pcVar3);
127             fclose(inputfile);
128             fclose(outputfile);
129             return 0xffffffff;
130           }
131           local_18 = fwrite(auStack1060,1,local_1c,outputfile);
132         } while ((int)local_1c <= (int)local_18);
133         printf("Error Writing - in %d - out %d\n",local_1c,local_18);
134         fclose(inputfile);
135         fclose(outputfile);
```

Motorola: Camera ←→ backend MQTT

- Camera authenticates using a client certificate
- Camera subscribes to /v1/devices/010335000AE2857FA7RVNGSQUA
  - Waits for commands (send through the app)
- Camera publishes to /v1/server/devices/010335000AE2857FA7RVNGSQUA
  - Publish responses to incoming commands

- We can dump/decrypt the firmware
  - We have access to the client certificate and private key
  - We can connect to the backend server as if we are the camera

Motorola: Camera ←→ backend MQTT

- Backend allows wildcard subscriptions /v1/devices/* and /v1/server/devices/*
  - We can see the messages exchanged by the server and **all** cameras
- Even worse: we can publish to any topic
  - Allows control over all cameras:
    - Rotate
    - Read temperature
    - Deactivate motion alerts
    - Play audio
    - Factory reset

Motorola: vulnerabilities

- We could connect to the main backend control server
- We could send arbitrary commands to all Motorola cameras
  - Rotate all cameras
  - Play audio on all baby monitors
  - Deactivate alerts
  - Factory reset all devices

Finding	Indicative severity	Description
F1	Low	Exposed debug interface allows to extract firmware
F2	Low	Hardcoded cryptographic keys for firmware update confidentiality
F3	Low/Medium	Local web interface exposes log files with sensitive information
F4	High	Stack based buffer overflow in local web server
F5	Critical	All devices use the same MQTT client certificate
F6	Critical	MQTT server allows wildcard subscriptions
F7	Critical	MQTT server allows any client to publish in any topic
F8	Low	Account enumeration through unauthenticated API endpoints

- **Attacking a single device leads to system wide compromise!**
- **You do not have to be the target!**

Car Key fobs

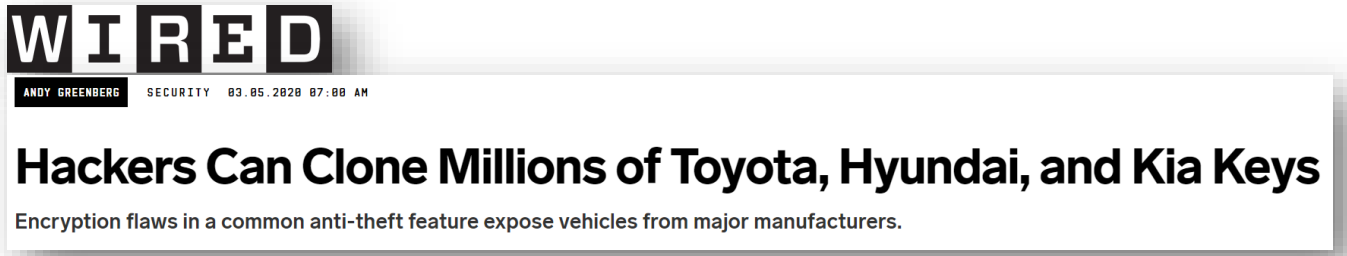

WIRED

ANDY GREENBERG SECURITY 09.10.18 01:00 PM

HACKERS CAN STEAL A TESLA MODEL S IN SECONDS BY CLONING ITS KEY FOB

40-bit proprietary crypto in cars made by Tesla, McLaren and Fisker.

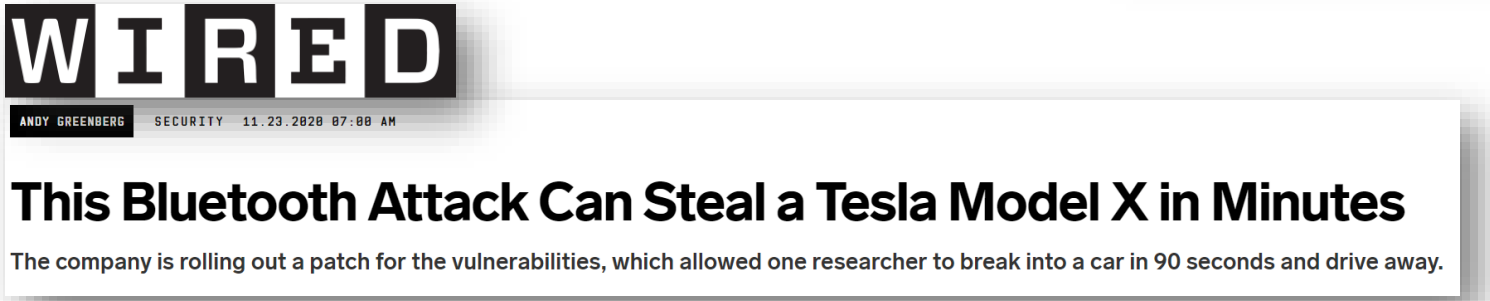
Used physical attacks to recover firmware from ECUs.
Found weak key derivation functions.
Cryptographic keys with up to 3 bytes of entropy.


WIRED

ANDY GREENBERG SECURITY 03.05.2020 07:00 AM

Hackers Can Clone Millions of Toyota, Hyundai, and Kia Keys

Encryption flaws in a common anti-theft feature expose vehicles from major manufacturers.


WIRED

ANDY GREENBERG SECURITY 11.23.2020 07:00 AM

This Bluetooth Attack Can Steal a Tesla Model X in Minutes

The company is rolling out a patch for the vulnerabilities, which allowed one researcher to break into a car in 90 seconds and drive away.

Exploiting a flaw in key fob firmware update and key fob pairing protocol allows to steal the car.

Fast, Furious and Insecure: Passive Keyless Entry and Start Systems in Modern Supercars: https://tches.iacr.org/index.php/TCHES/article/view/8289
Dismantling DST80-based immobiliser systems: https://tches.iacr.org/index.php/TCHES/article/view/8546
My other car is your car: compromising the Tesla Model X keyless entry system: https://tches.iacr.org/index.php/TCHES/article/view/9063

Remote Keyless Entry

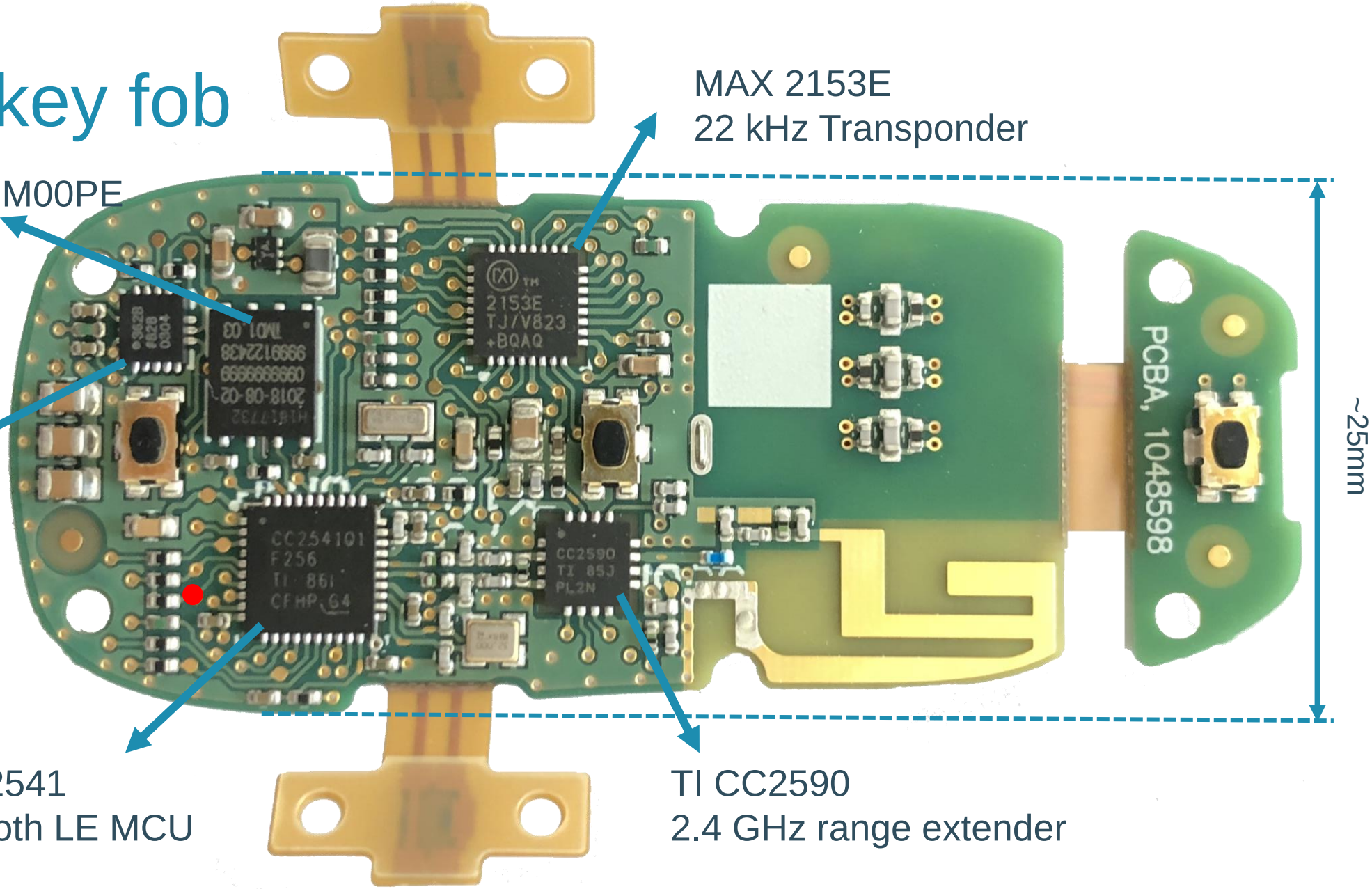
rolling/hopping code

Passive Keyless Entry

challenge

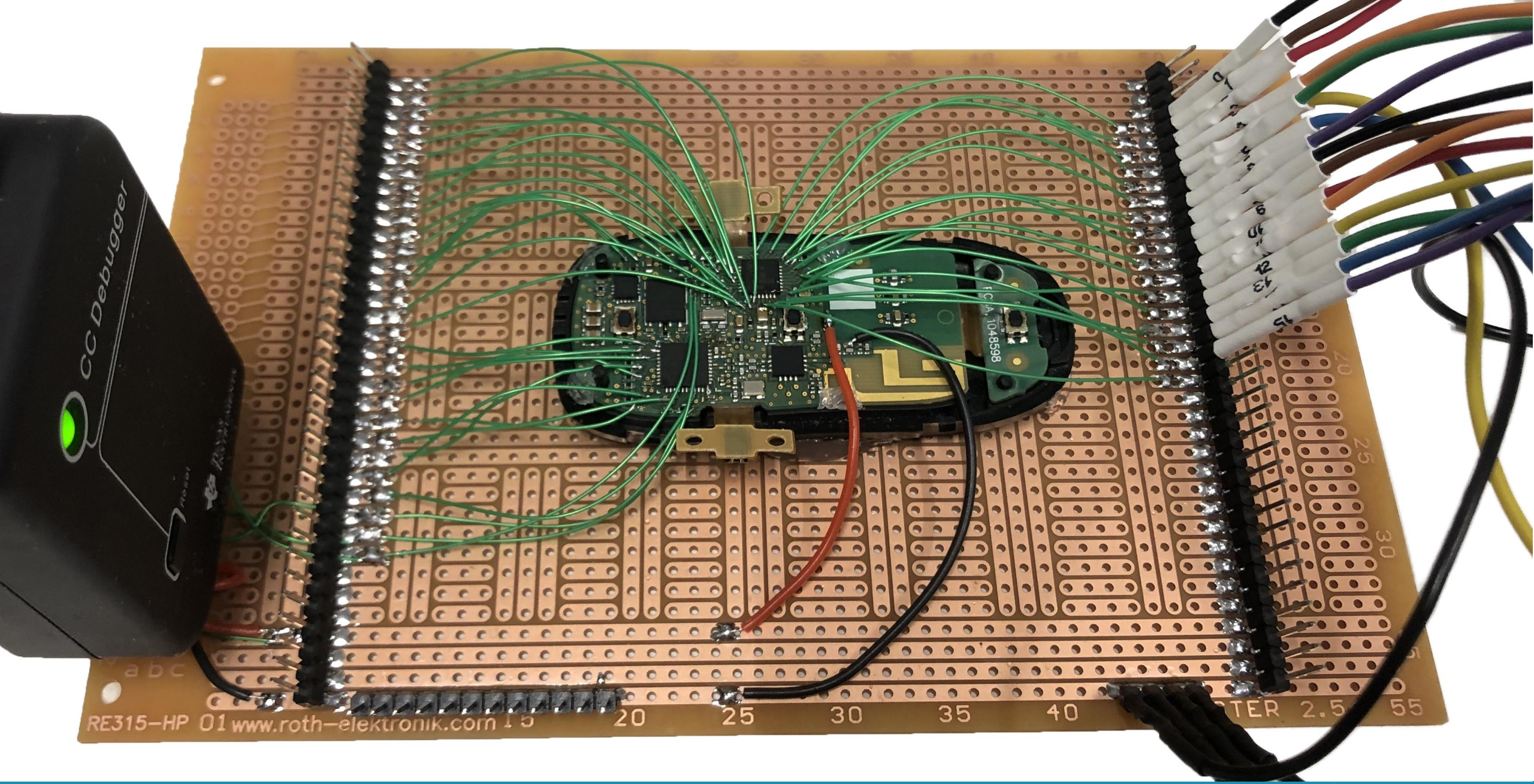
response

Model X key fob



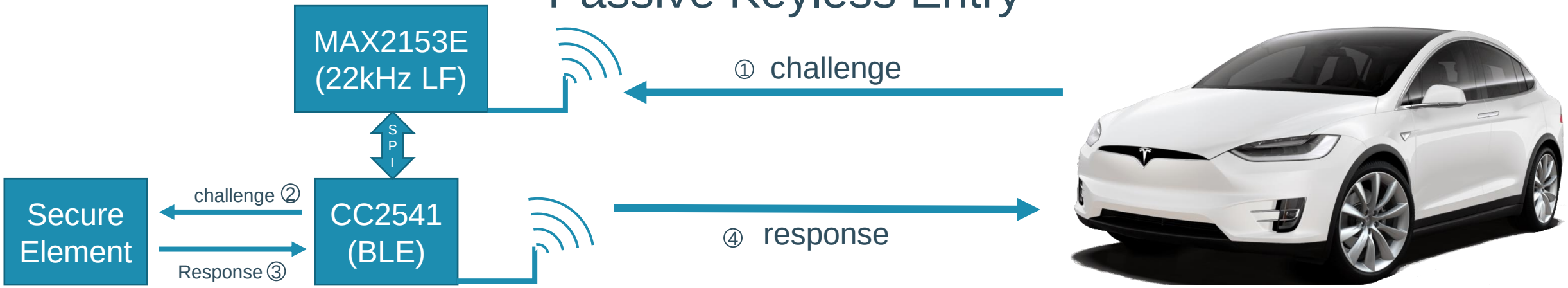
COSIC

KU LEUVEN


CC Debugger
RE315-HP 01 www.roth-elektronik.com
20
25
30
35
40

Remote Keyless Entry

Passive Keyless Entry



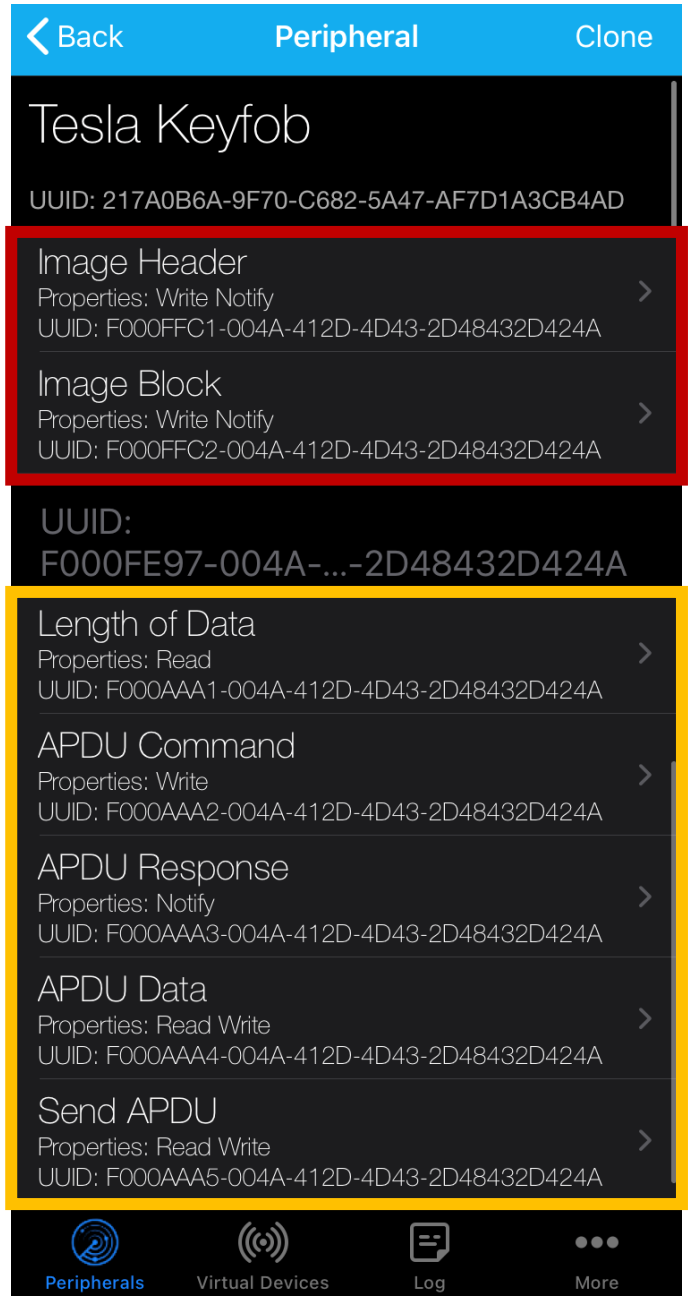
COSIC

KU LEUVEN

BLE Interface

- Key fob is a BLE peripheral
- Reseat the battery (power cycle)
  - Key fob advertises as connectable

- Over-Air-Download (OAD)
- Application Protocol Data Unit (APDU) Interface
  - Allows to send APDU commands to the SE over BLE

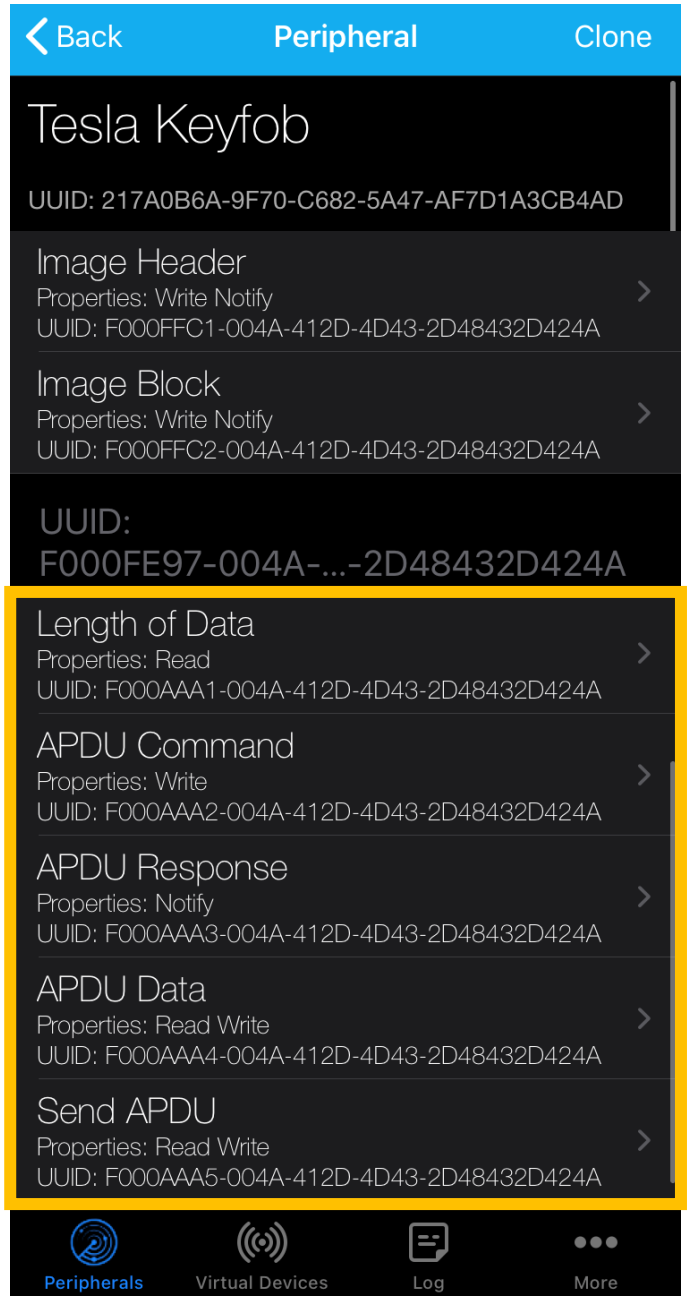


COSIC

KU LEUVEN

CC2541: BLE to APDU interface

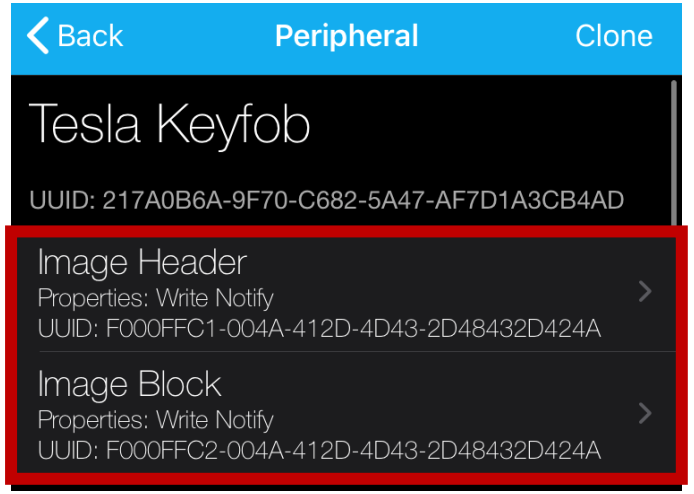
- Interface with the secure element via the CC2541
- Functionality that is used during pairing
- Interface has a block list implemented
  - “get unlock token” is blocked



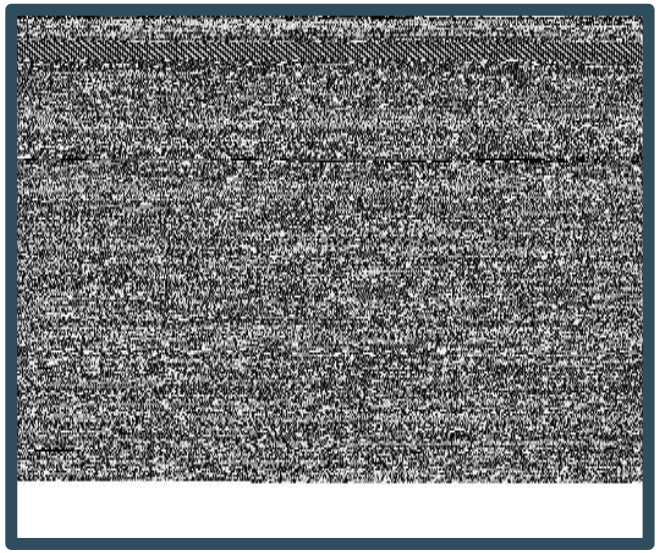
COSIC

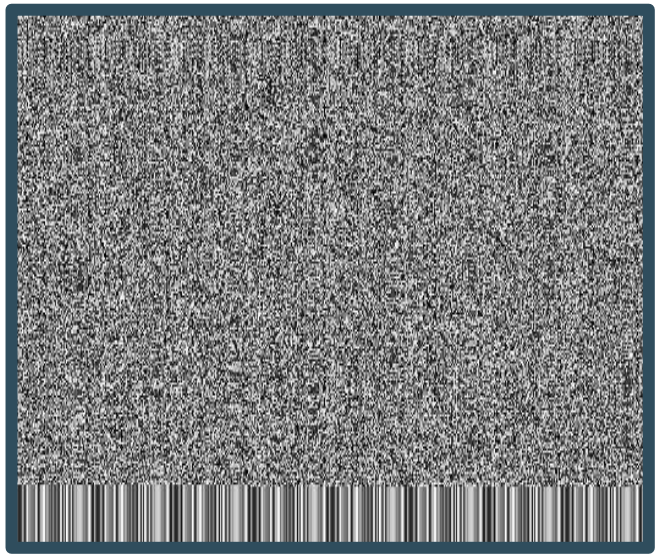
KU LEUVEN

CC2541: Over-Air-Download



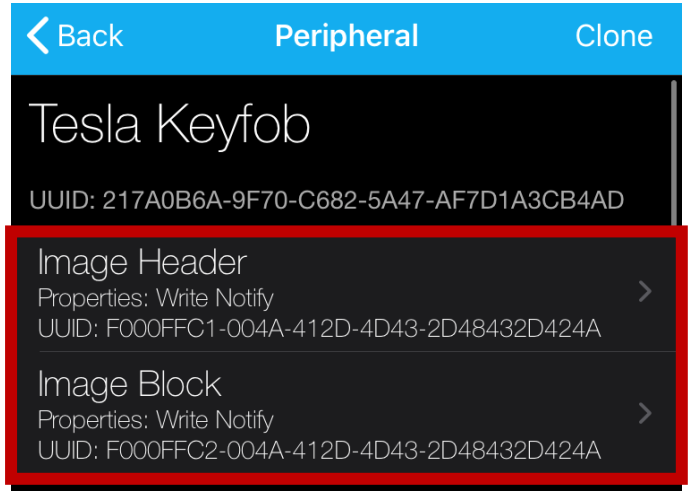
- Allows to update the CC2541 firmware over BLE
- OAD example implementation vulnerabilities:
  - AES-CTR: counter reset after 4 AES blocks
  - CBC-MAC: using the same key and non-constant time memcmp





https://e2e.ti.com/support/wireless-connectivity/bluetooth-group/bluetooth/f/bluetooth-forum/884322/faq-cc254x-oad-aes-ctr-crypto-implementation-vulnerability
https://e2e.ti.com/support/wireless-connectivity/bluetooth-group/bluetooth/f/bluetooth-forum/884316/faq-cc254x-oad-aes-cbc-mac-verification-vulnerability

CC2541: Over-Air-Download



- Allows to update the CC2541 firmware over BLE

- Tesla did not use the security components provided by the example
  - Plaintext firmware, with a SHA-1 hash and RSA signature
  - RSA signature verification by secure element
    - SHA-256 over the SHA-1 hash, RSA-PSS
  - Result is ignored…
    - Leftover from development?

CC2541: Malicious firmware

- Goal: firmware that allows to send arbitrary APDU commands

1. Custom firmware
   - Requires some familiarity with TI tooling and SDK
   - Likely reduced functionality of the key fob
2. Static reverse engineering
   - No good support for our platform (customized 8051)
   - Can be time consuming and tedious
3. Brute force
   - Easily automated

CC2541: Brute force firmware modification

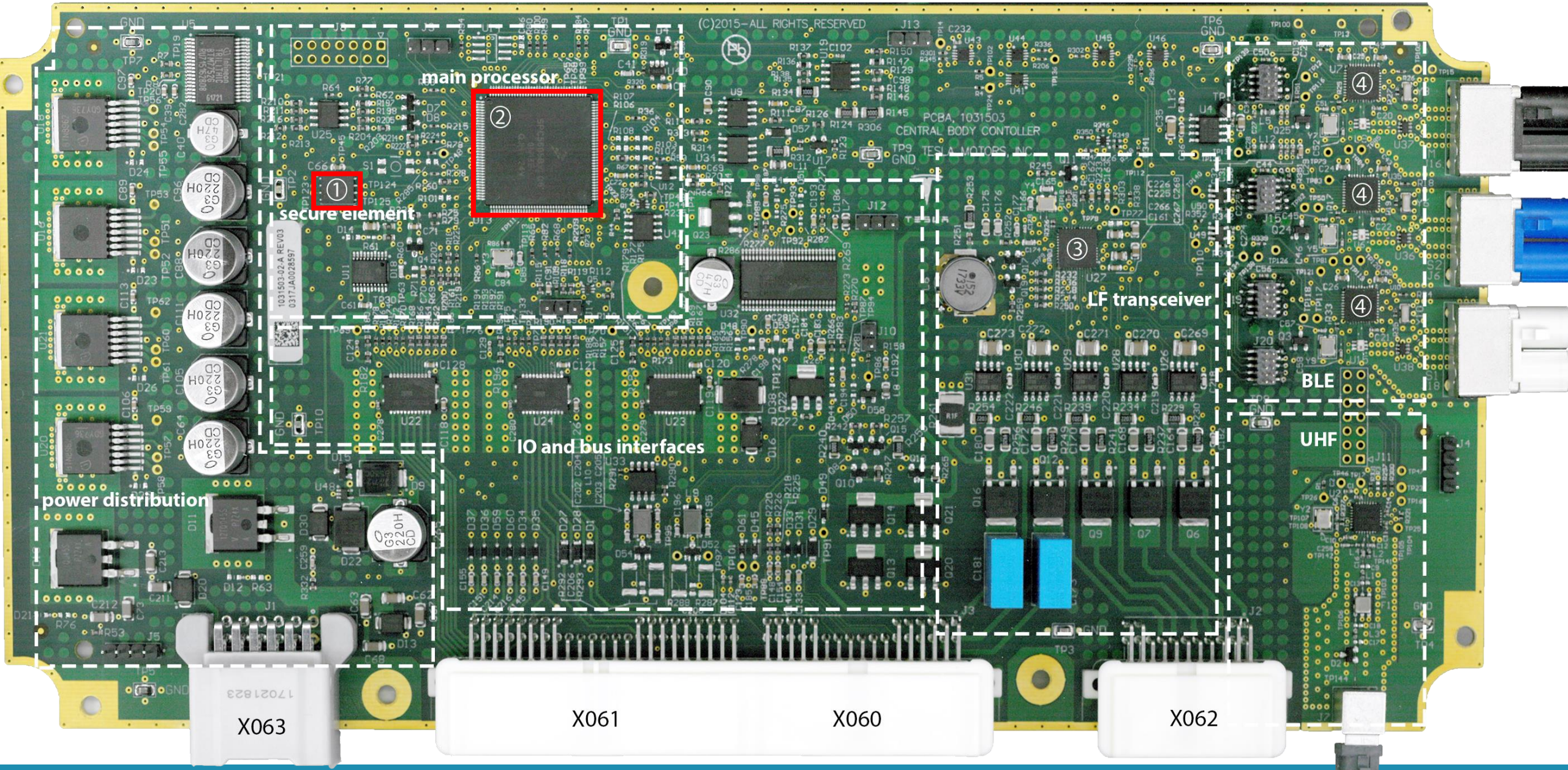
- Assumption: block list is implemented using *if* statements or *case*
- Likely uses conditional instructions
  - Jump if Accumulator Not Zero (JNZ) – 0x70
  - Jump if Accumulator Zero (JZ) – 0x60

- Replace occurrence of 0x70 (JNZ) with 0x60 (JZ)
- Flash the firmware
- Connect over BLE and try to send APDU
- Repeat until it works

Automated!

Plan of attack

1. Reseat the key fob battery
2. Connect to the key fob and push malicious firmware
   - Firmware is modified to allow all APDUs
3. Request a valid rolling code through the BLE APDU interface
4. Use the acquired rolling code to unlock the car
   - This code can only be used to unlock the car

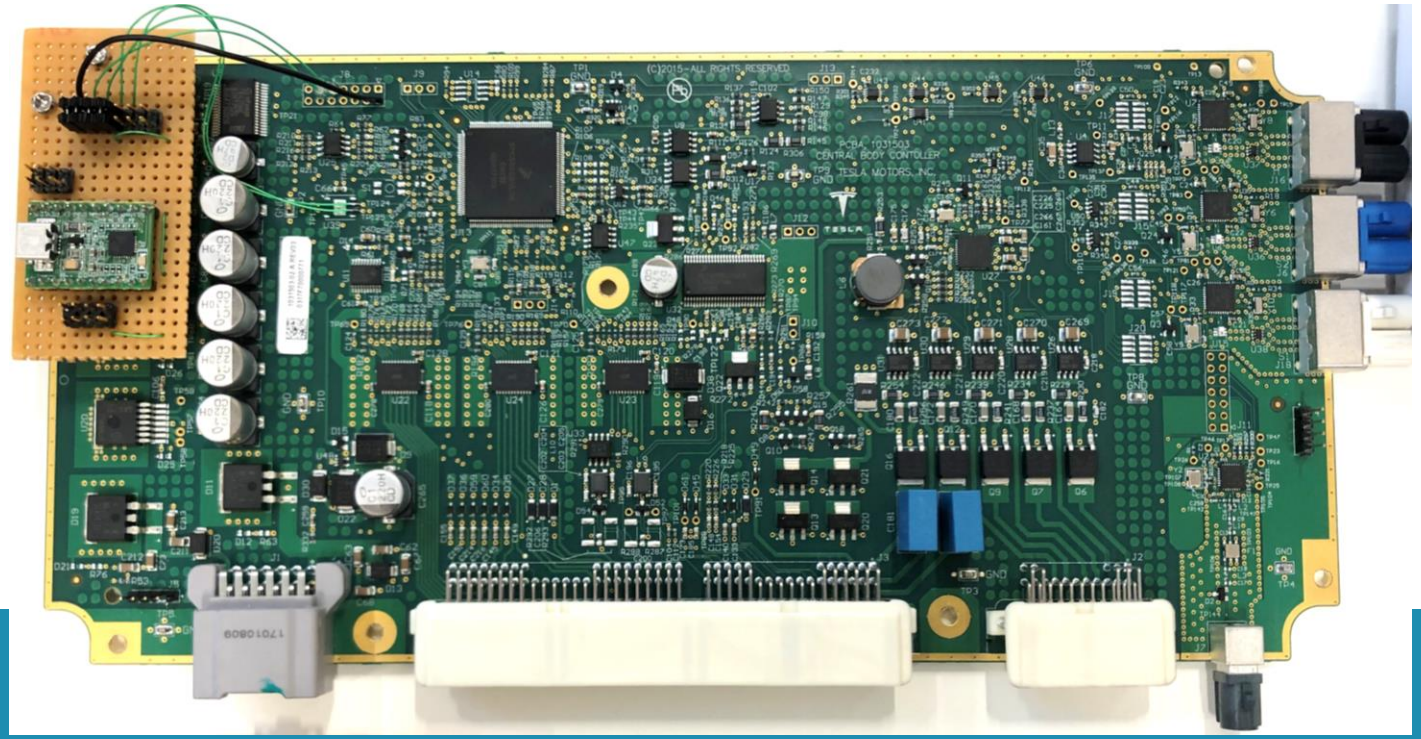


COSIC

KU LEUVEN

Force key fob wake-up

- Body Control Module can send a wake-up command over LF
- Allows a car to wake-up key fobs that have been paired to it
  - Based on an identifier derived from the VIN
  - VIN is readable from the windshield

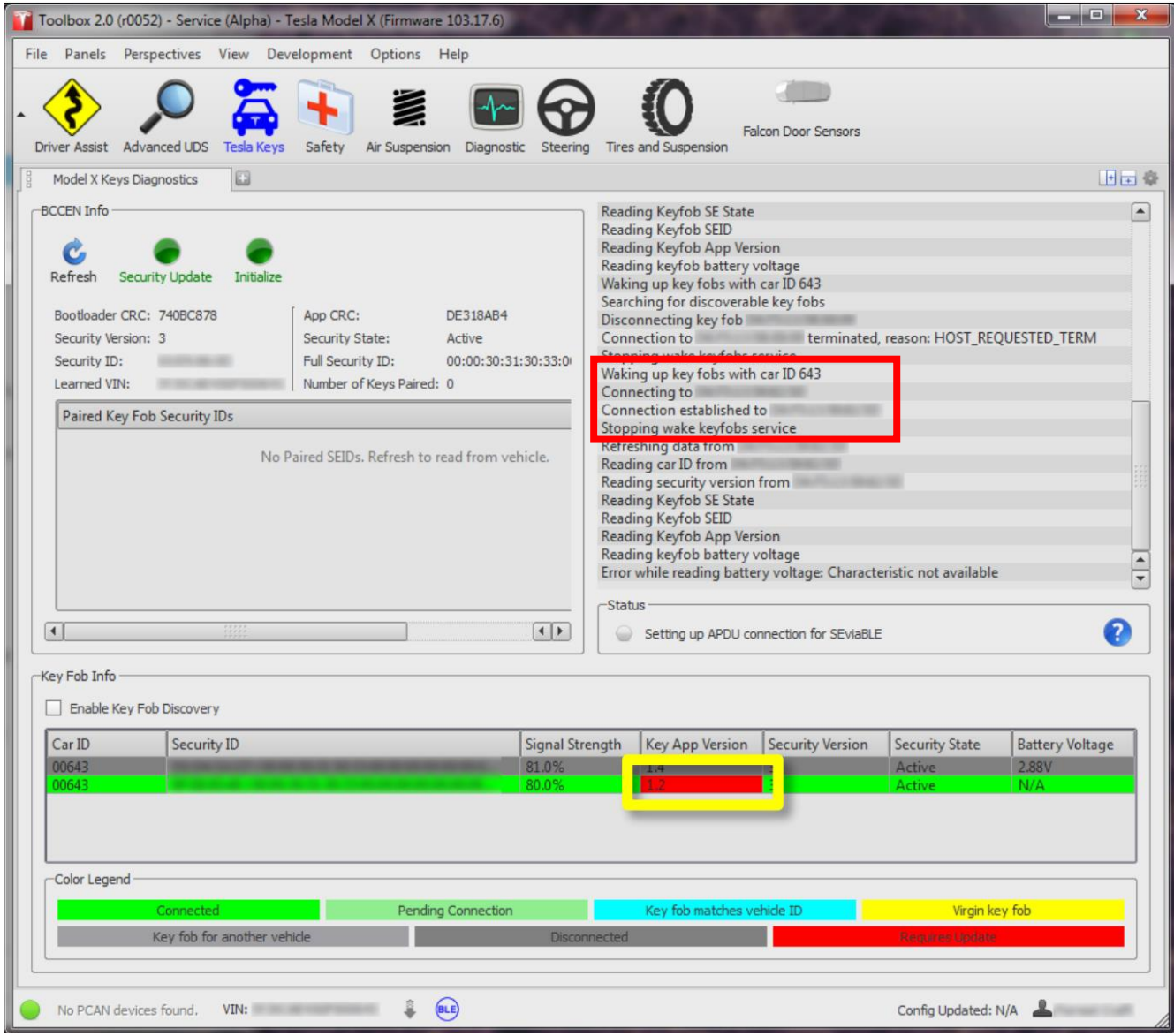


Plan of attack

1. Request target key fob to advertise as connectable
   - Can be achieved using a modified BCM
2. Connect to the key fob and push malicious firmware
   - Firmware is modified to allow all APDUs
3. Request a valid rolling code through the BLE APDU interface
4. Use the acquired rolling code to unlock the car
   - This code can only be used to unlock the car

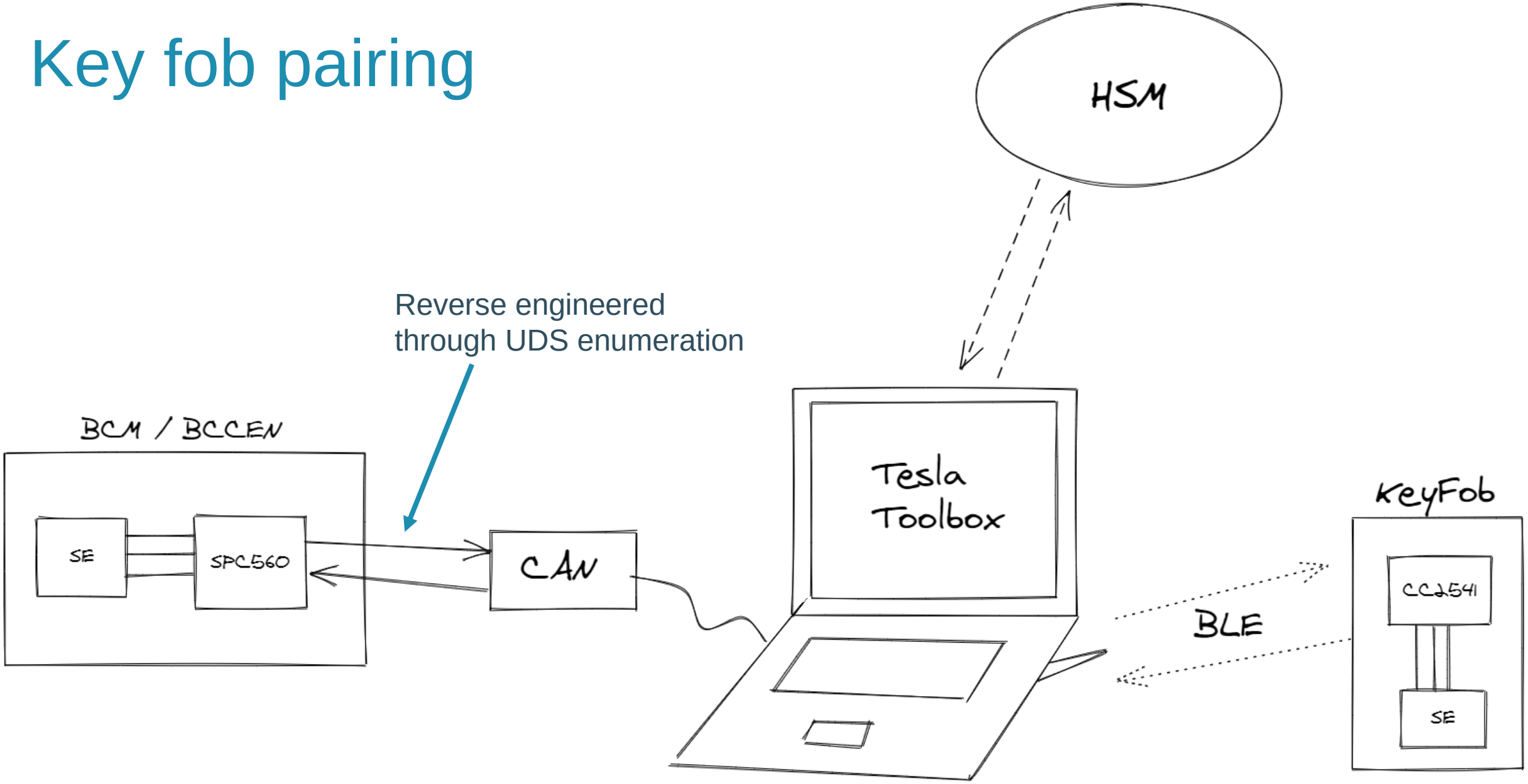
Toolbox

- Used for servicing Model S and X
- Not publicly available
  - Available ‘on the internet’
  - Briefly (unintentionally?) released
- Interesting parts are stored encrypted
  - must be decrypted before use ;)



Picture source: https://static.nhtsa.gov/odi/tsbs/2016/SB-10081823-5448.pdf
Unintentional Toolbox release: https://twitter.com/greentheonly/status/1334564079264993281

Key fob pairing



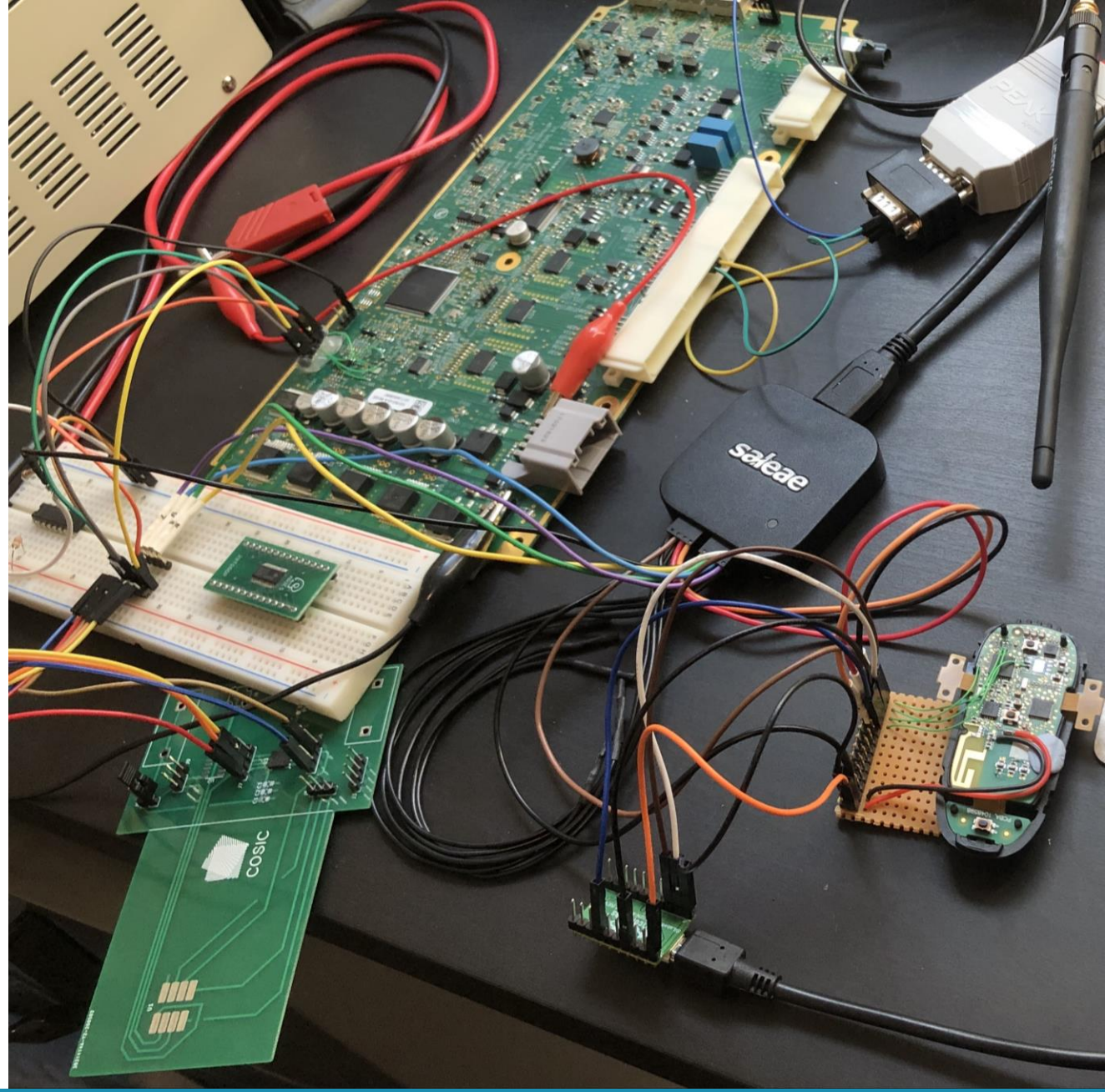
COSIC

KU LEUVEN

UDS enumeration

- ISO14229

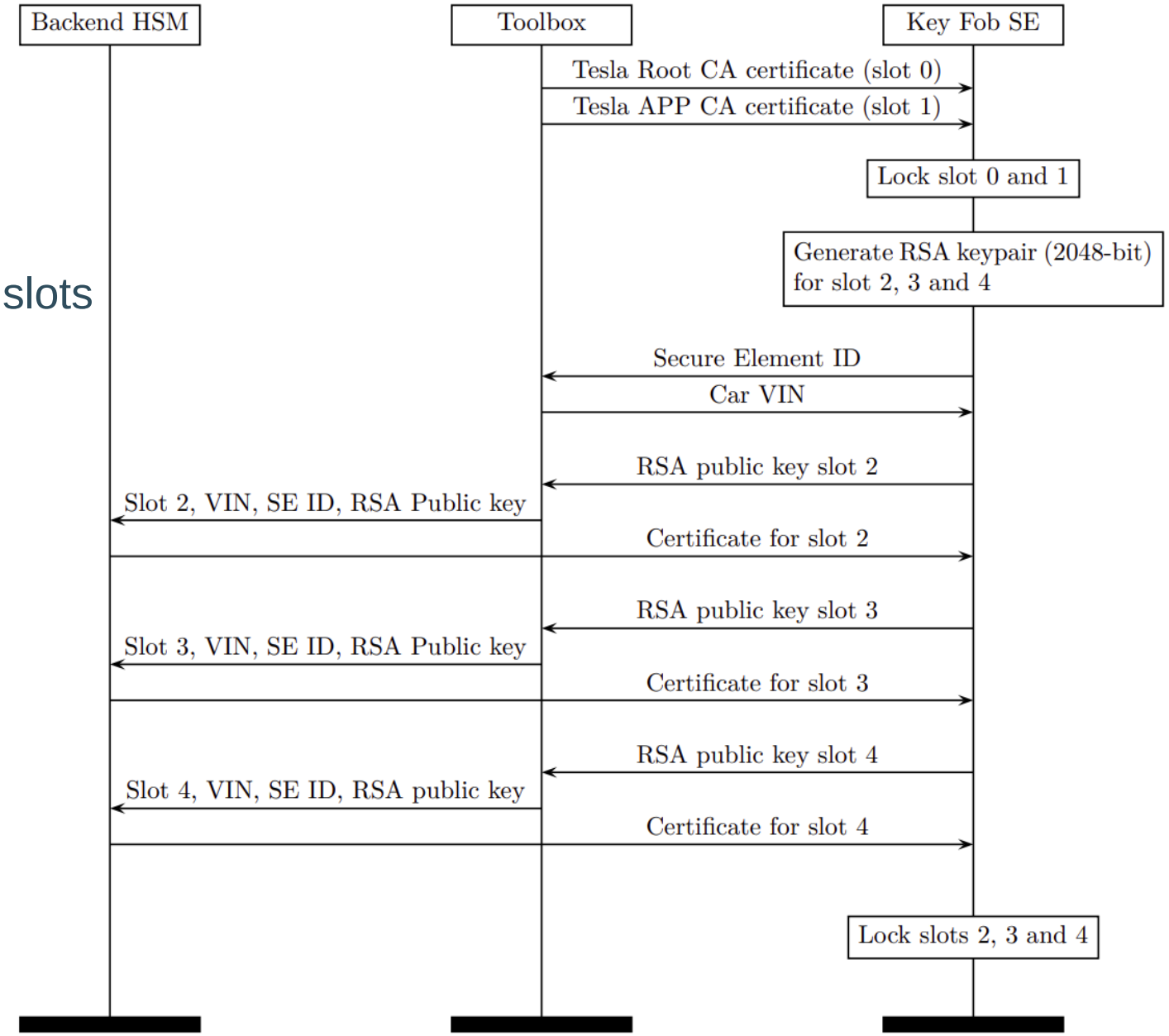
1. enumerate existing routines
2. Enumerate option record length
3. Start routines with option record
4. Did my data arrive at the SE?



Provisioning

- Key fob Secure Element (SE) has 5 RSA slots
  - Slot 0 and 1: Tesla CA certificates
  - Slot 2, 3 and 4: key fob specific

- HSM signs certificate for slot 2, 3 and 4
  - Presumably so the car can ensure it is pairing to a legitimate key fob
  - Certificates are stored in key fob SE

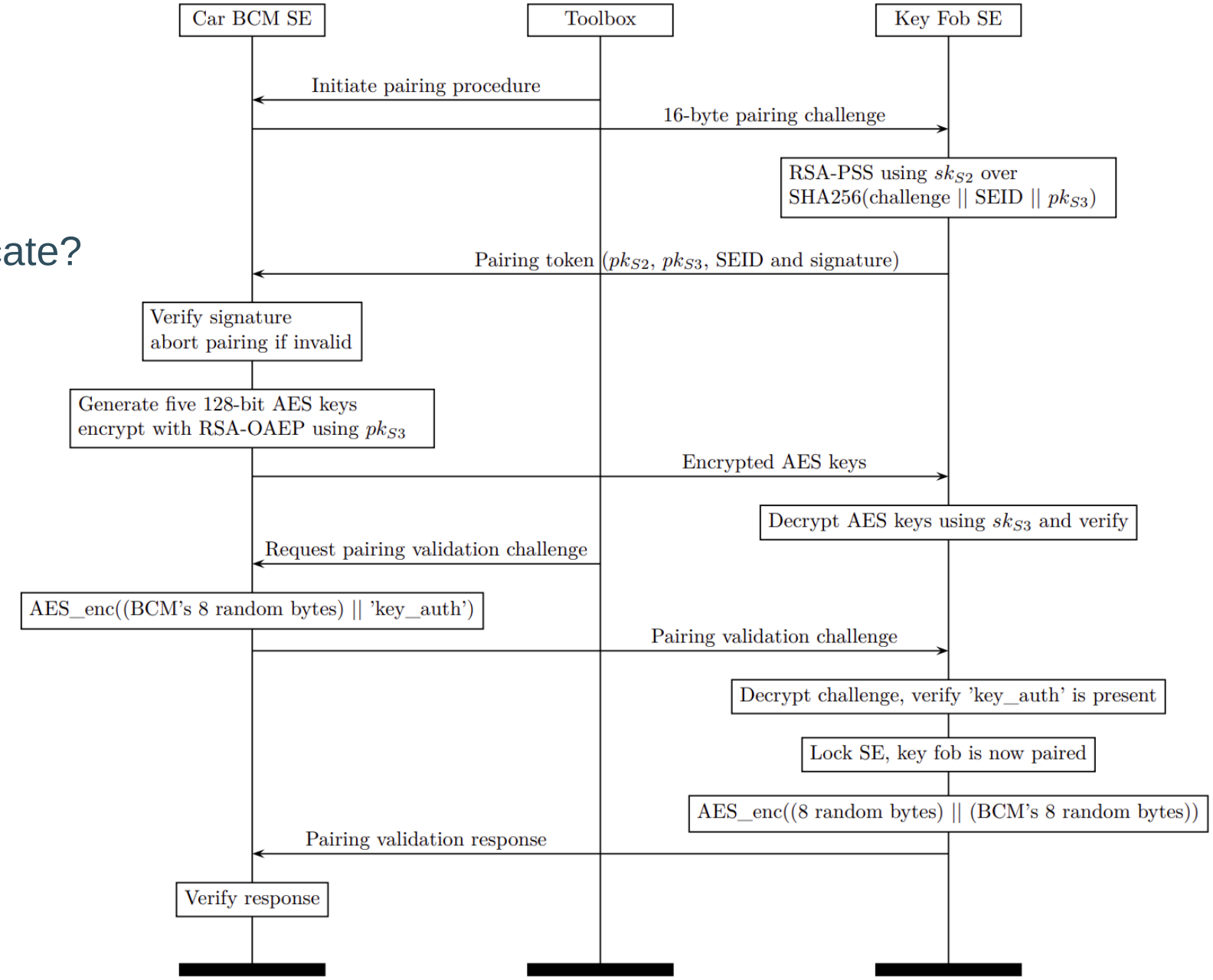


COSIC

KU LEUVEN

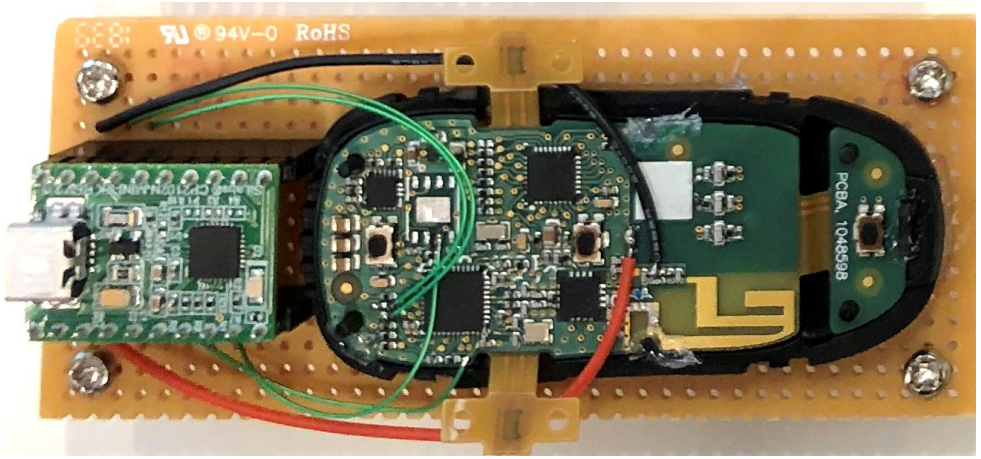
Pairing

- Where's ~~Wally/Waldo~~ the certificate?



Modified keyfob

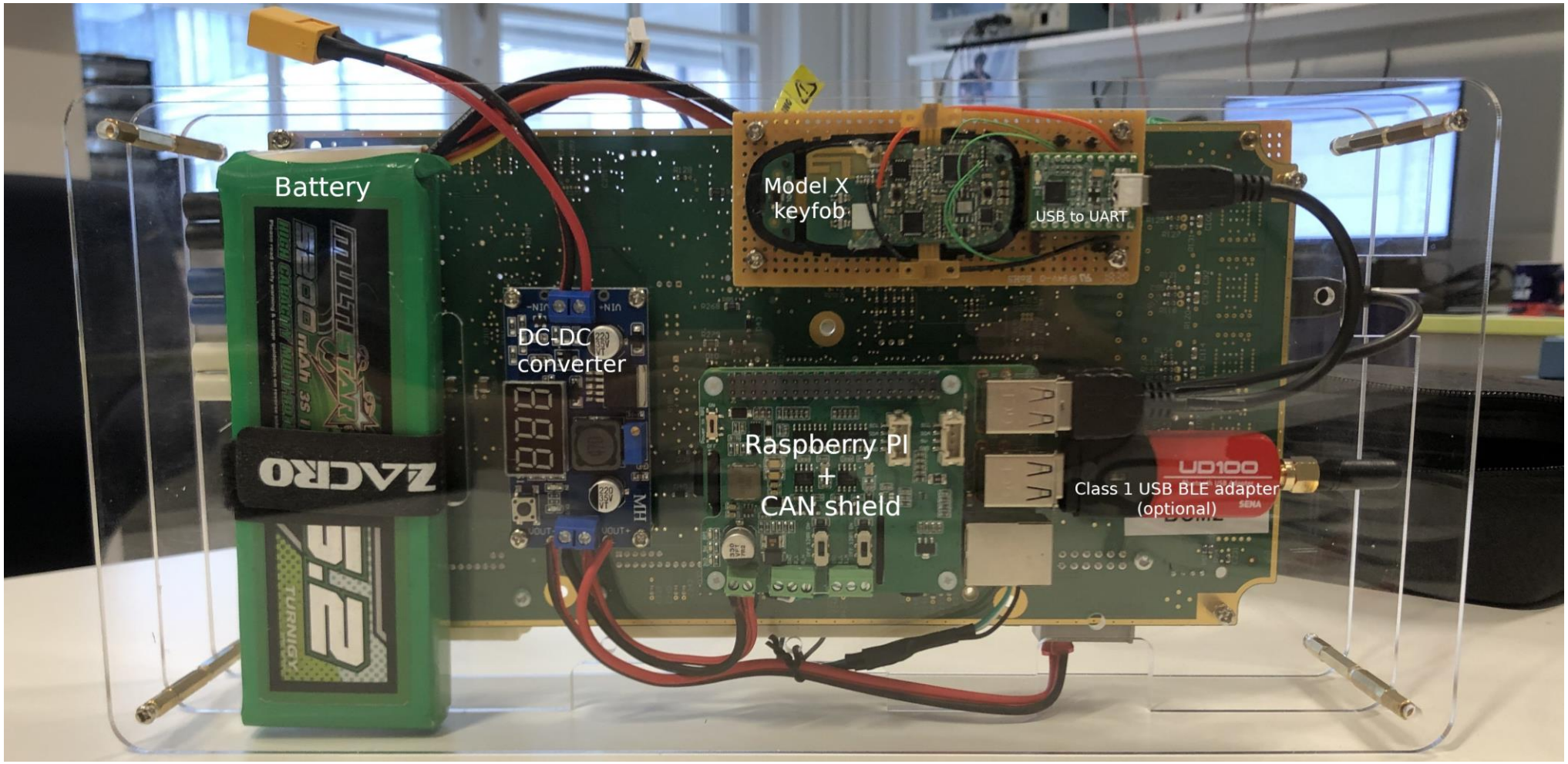
- Replace the secure element
  1. Matching footprint component and low(er) level programming
  2. USB to UART and Python!
     - CP2102N-MINIEK
       - Arbitrary baud rate
       - Extra GPIO (for RST)



Plan of attack

1. Request target key fob to advertise as connectable
   - Can be achieved using a modified BCM
2. Connect to the key fob and push malicious firmware
   - Firmware is modified to allow all APDUs
3. Request a valid rolling code through the BLE APDU interface
4. Use the acquired rolling code to unlock the car
   - This code can only be used to unlock the car
5. Connect to the diagnostic port and pair a modified key fob to the car
   - Key fob is modified by replacing the secure element

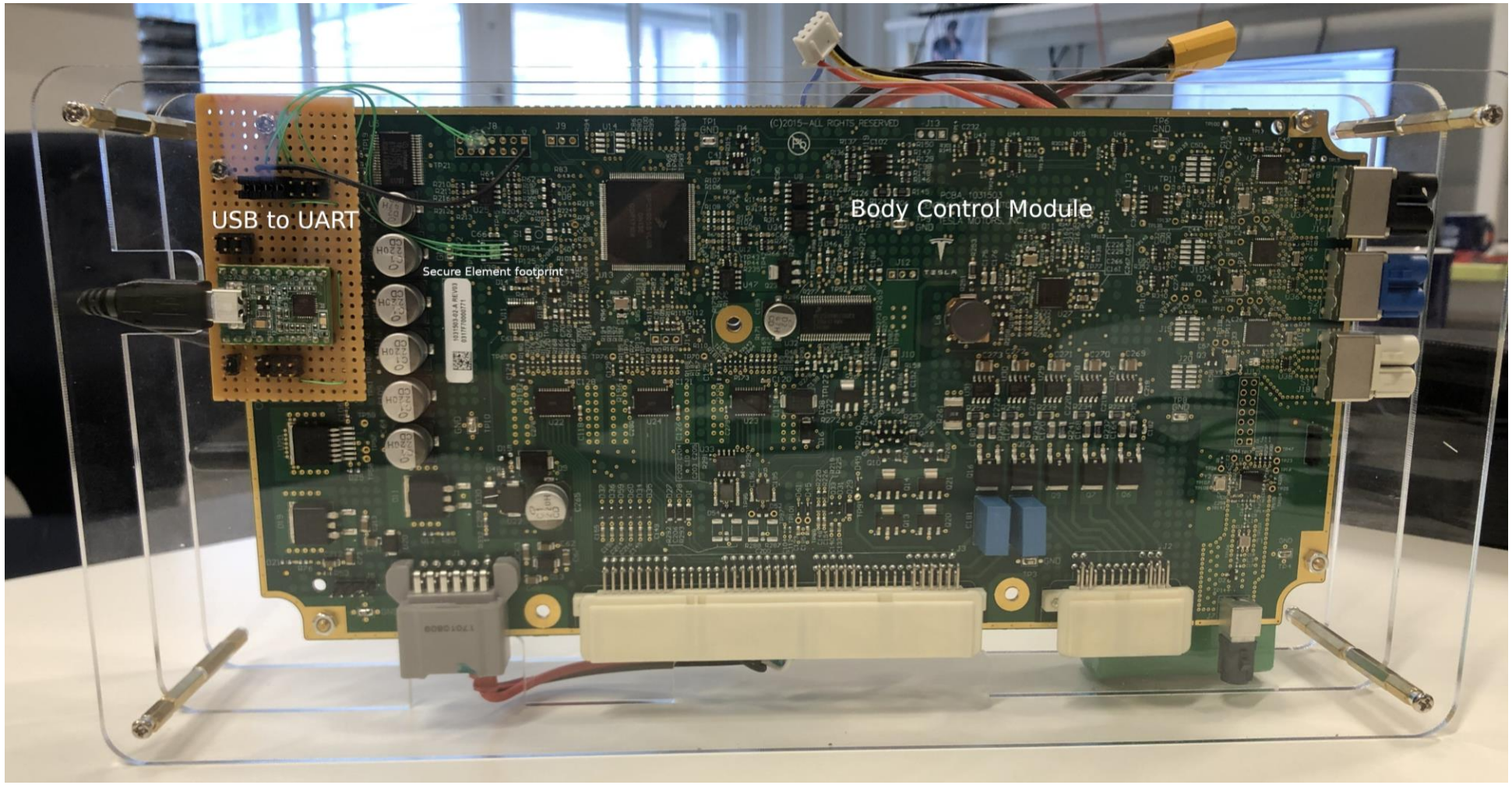
Proof of Concept



COSIC

KU LEUVEN

Proof of Concept



COSIC

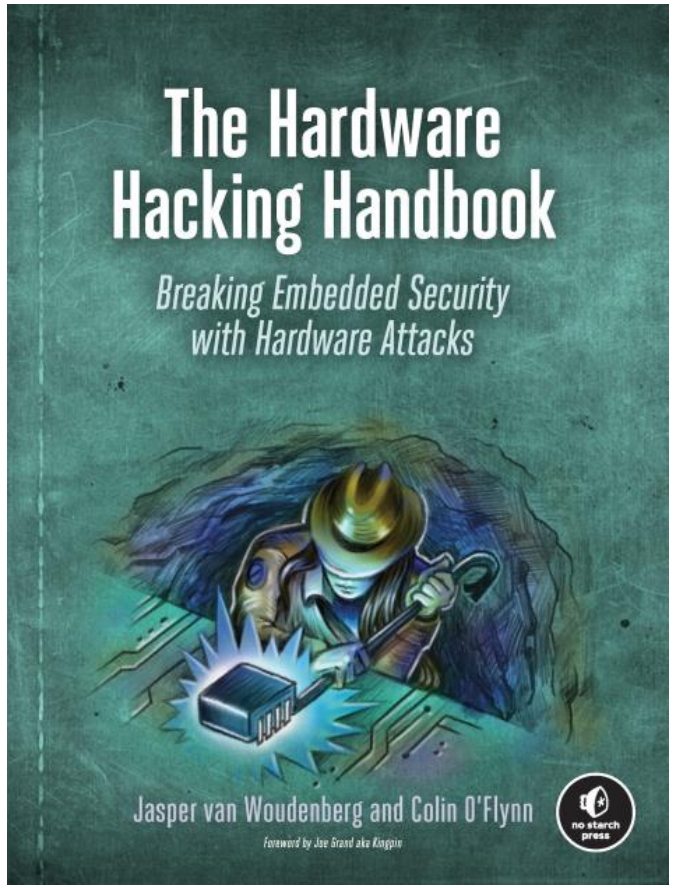
KU LEUVEN


COSIC
https://youtu.be/watch?v=clrNuBb3myE

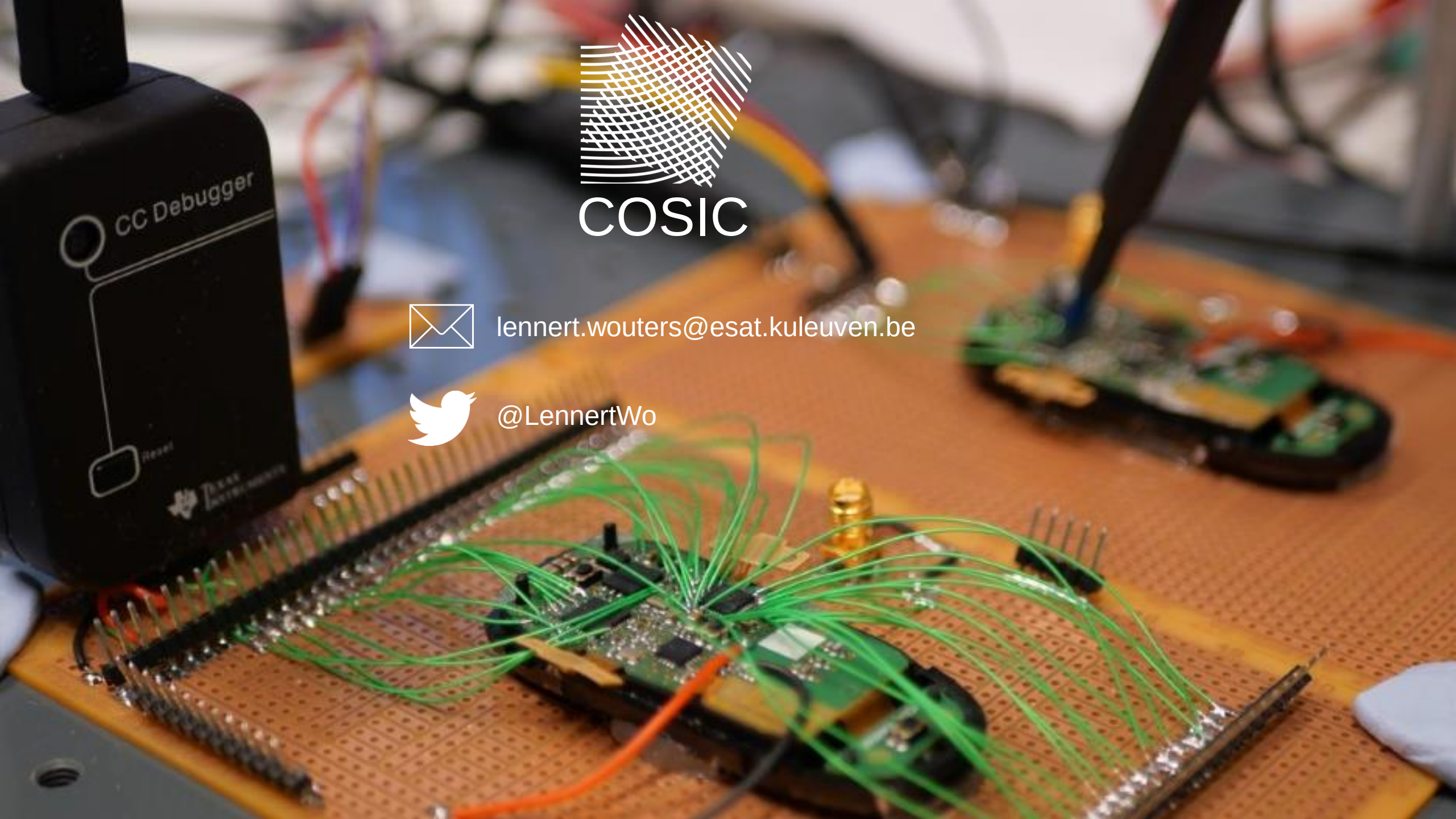
Conclusions

- Embedded devices are often exposed to physical attackers
  - (Physical) attackers challenge common assumptions
- Many countermeasures exist!
  - Difficult cost benefit tradeoff

- Do you have an unused device at home?
  - Have a look inside ☺



https://nostarch.com/hardwarehacking



COSIC

lennert.wouters@esat.kuleuven.be

@LennertWo