Glitched on Earth by Humans: A Black-Box Security Evaluation of the SpaceX Starlink User Terminal

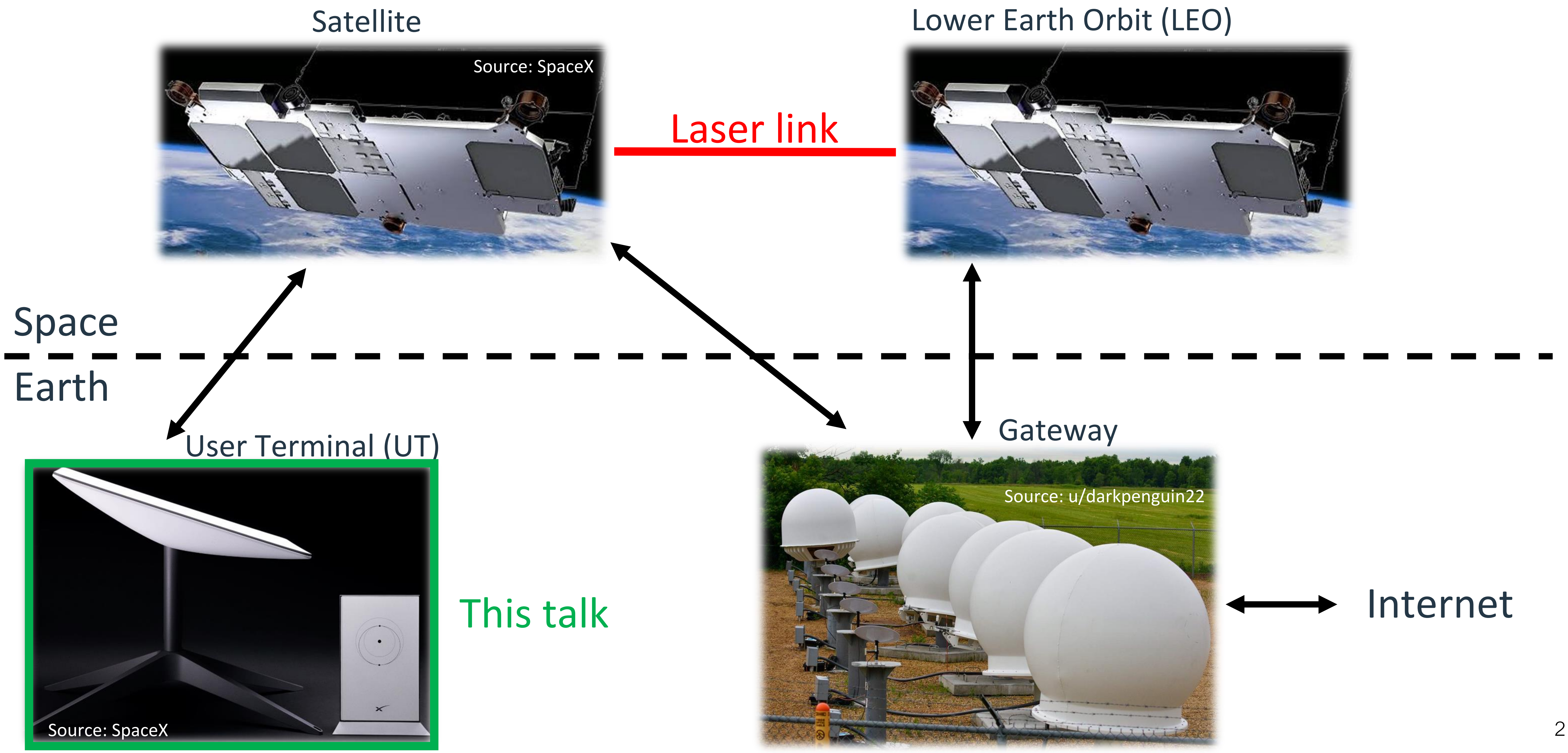
Lennert Wouters

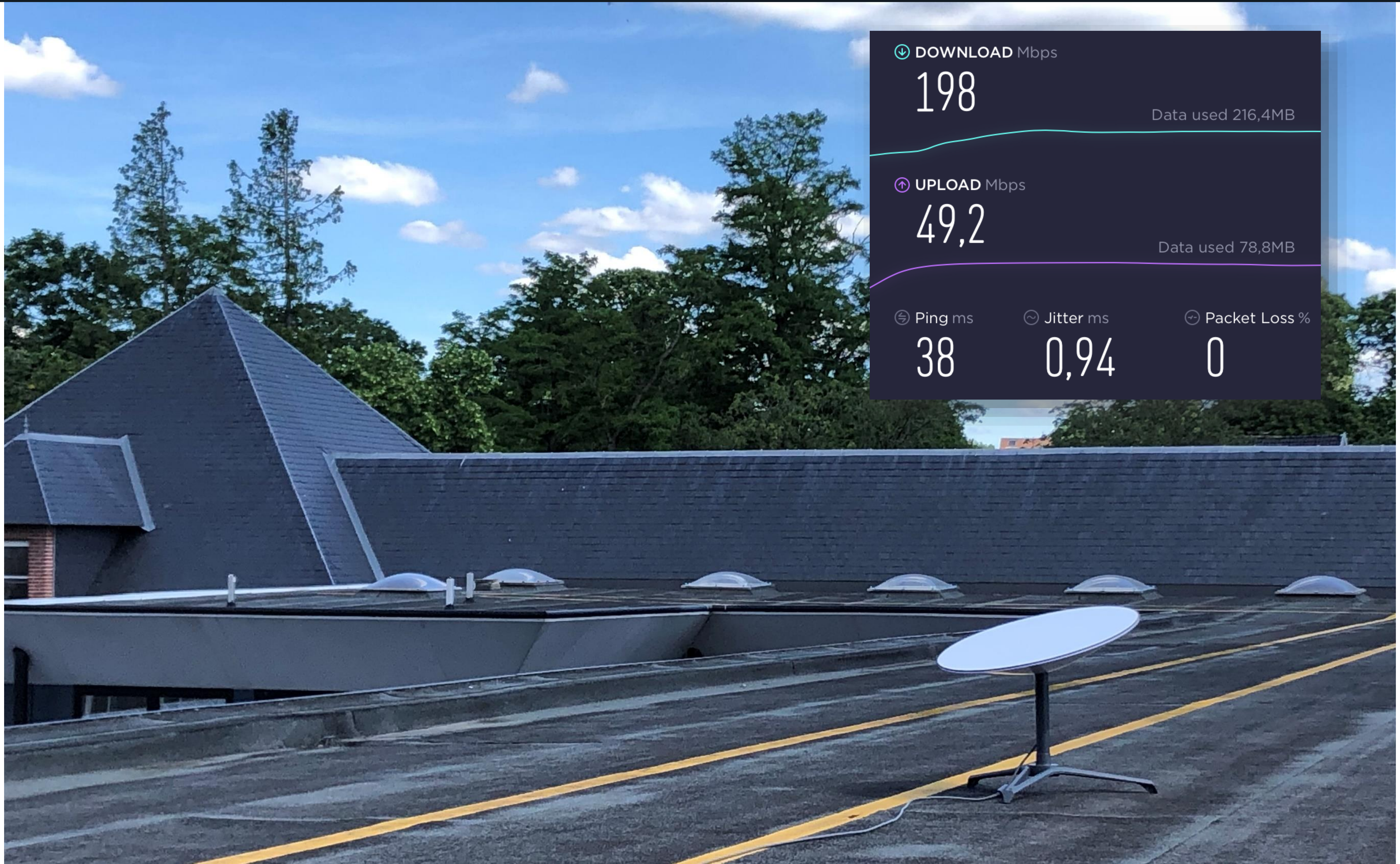
@LennertWo

COSIC

KU LEUVEN

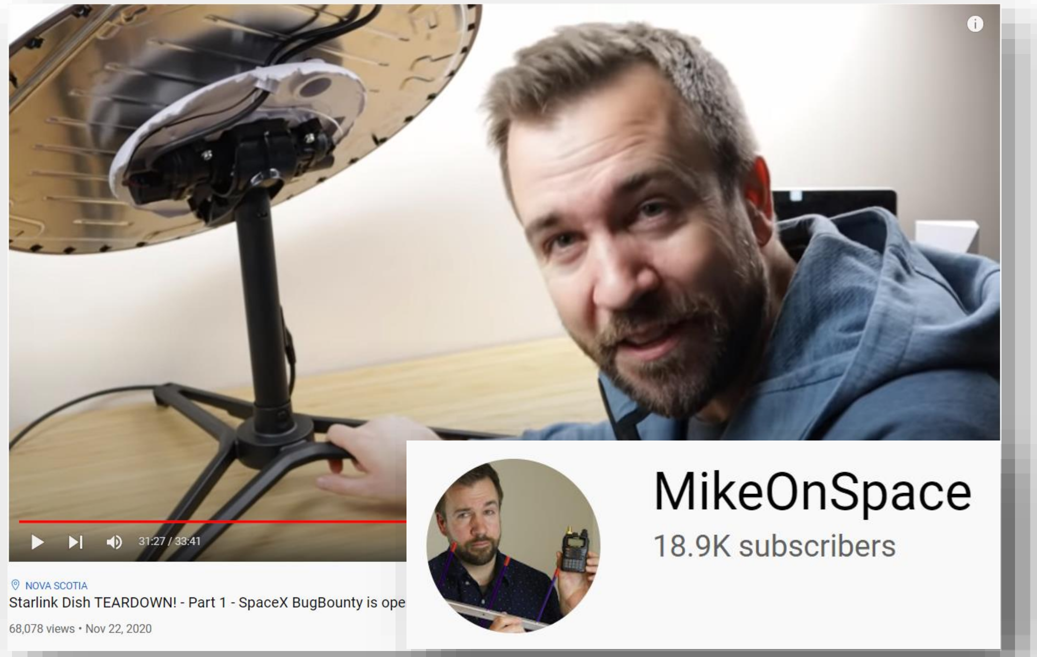
Starlink 101





DOWNLOAD Mbps
198
Data used 216,4MB
UPLOAD Mbps
49,2
Data used 78,8MB
Ping ms
38
Jitter ms
0,94
Packet Loss %
0

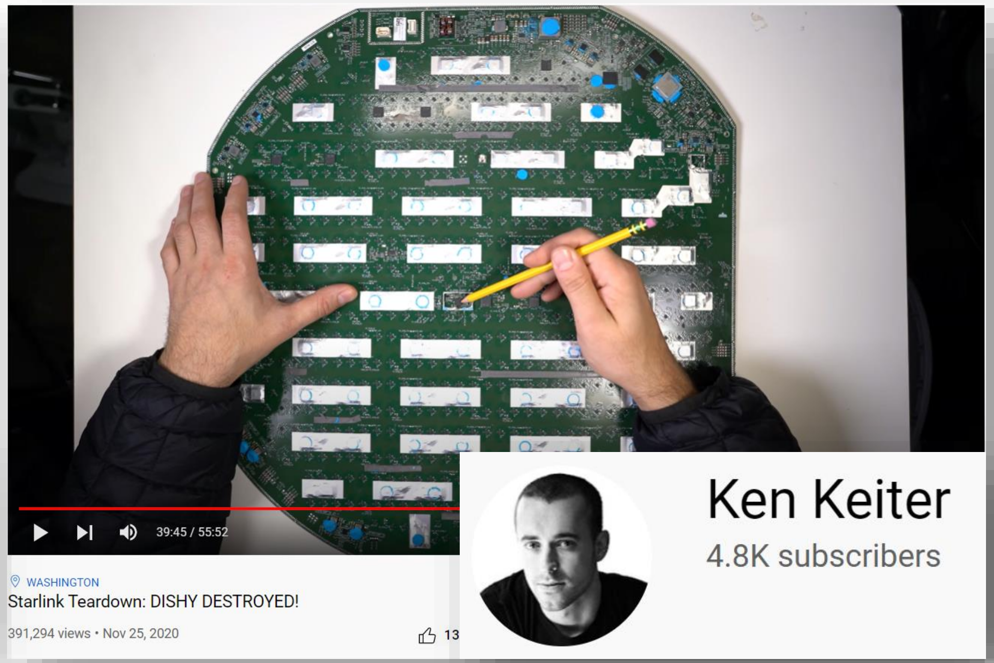
Teardowns



NOVA SCOTIA
Starlink Dish TEARDOWN! - Part 1 - SpaceX BugBounty is ope
68,078 views • Nov 22, 2020

MikeOnSpace
18.9K subscribers

youtube.com/c/MikeOnSpace @mikeonspace



WASHINGTON
Starlink Teardown: DISHY DESTROYED!
391,294 views • Nov 25, 2020

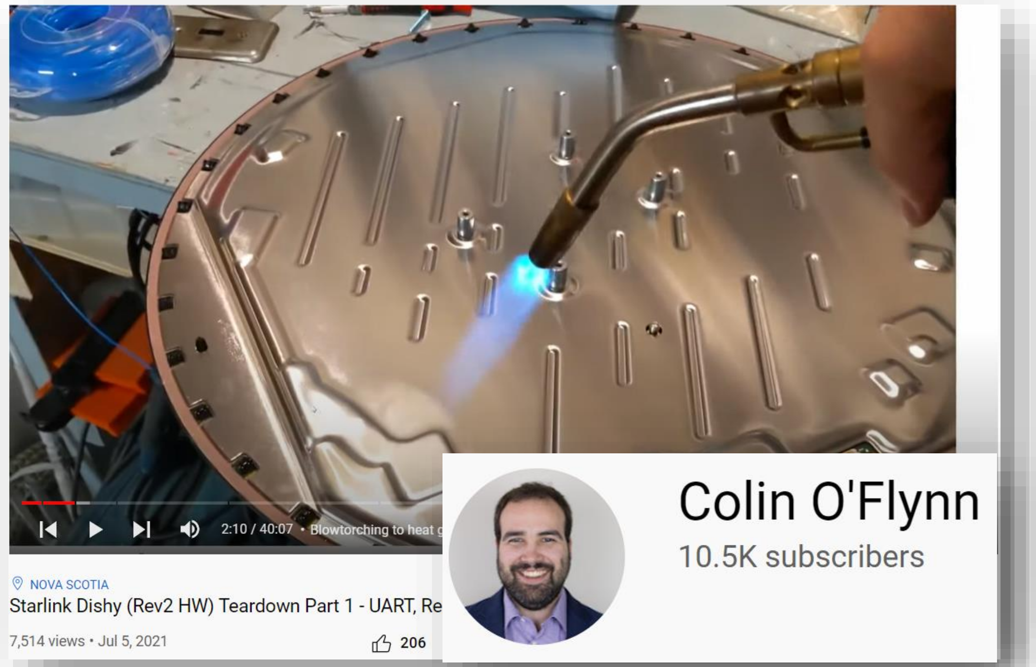
Ken Keiter
4.8K subscribers

youtube.com/c/KenKeiter @kenkeiter

TSP #181 - Starlink Dish Phased Array Design, Architectu
102,965 views • Jan 11, 2021

The Signal Path
102K subscribers

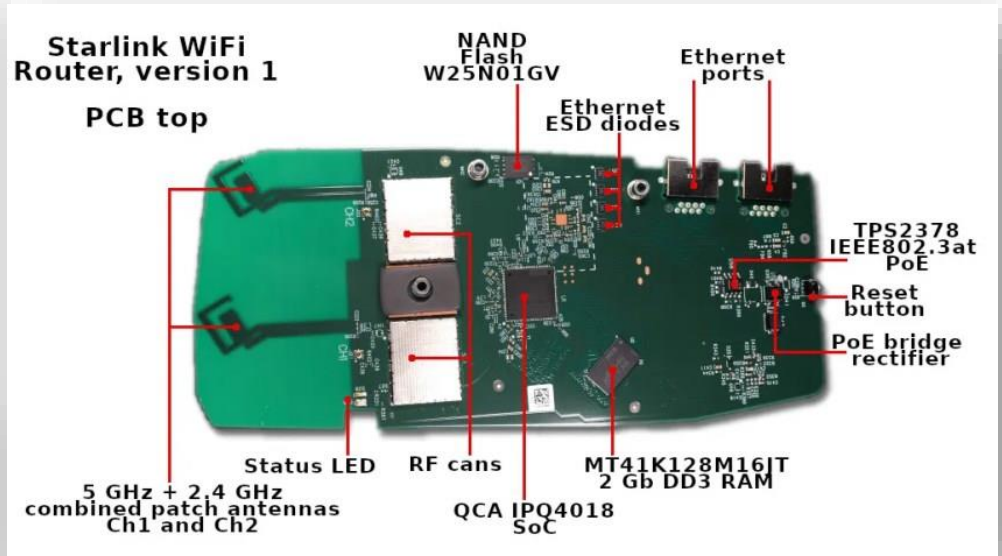
youtube.com/c/Thesignalpath @TheSignalPath



NOVA SCOTIA
Starlink Dishy (Rev2 HW) Teardown Part 1 - UART, Re
7,514 views • Jul 5, 2021

Colin O'Flynn
10.5K subscribers

youtube.com/c/ColinOFlynn @colinoflynn



olegkutkov.me @olegkutkov

danmurray.net
@DanJMurray

Hardware revisions

Circular UT

- 59 cm (23,23”) diameter
- Residential

- rev1_pre_production
- rev1_production
- rev1_proto1/2/3
- rev2_proto0/1/3
- rev2_proto2 (SoC cut 3)
- rev2_proto4 (SoC cut 4)

Square UT

- 50 x 30 cm (19” x 12”)
- Residential and RV

- rev3_proto0
- rev3_proto1
- rev3_proto2

- rev4_proto1
- mini1_

High Performance UT

- 57 x 51 cm (22” x 20”)
- Business and Maritime

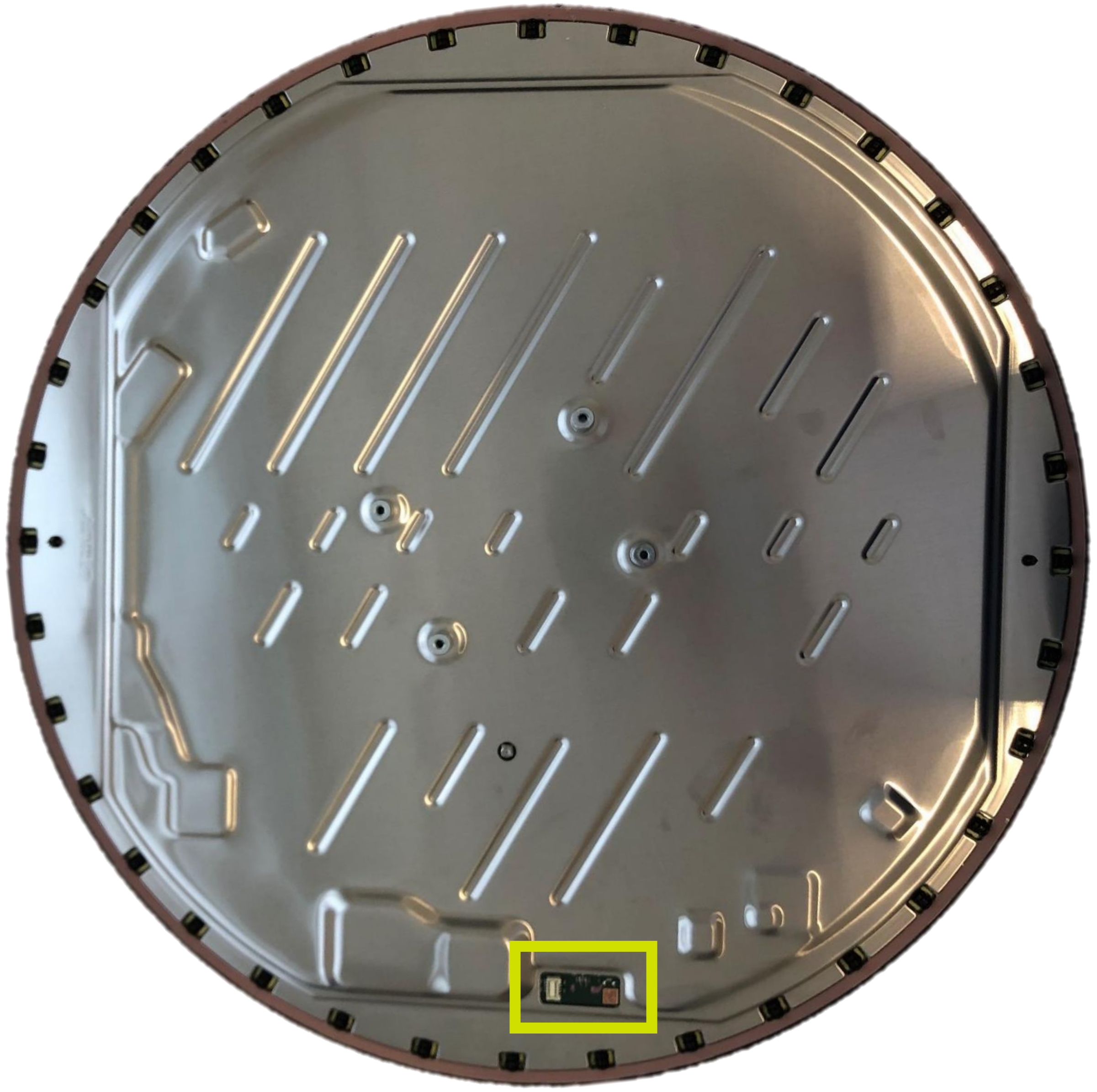
- hp1_proto0
- hp1_proto1

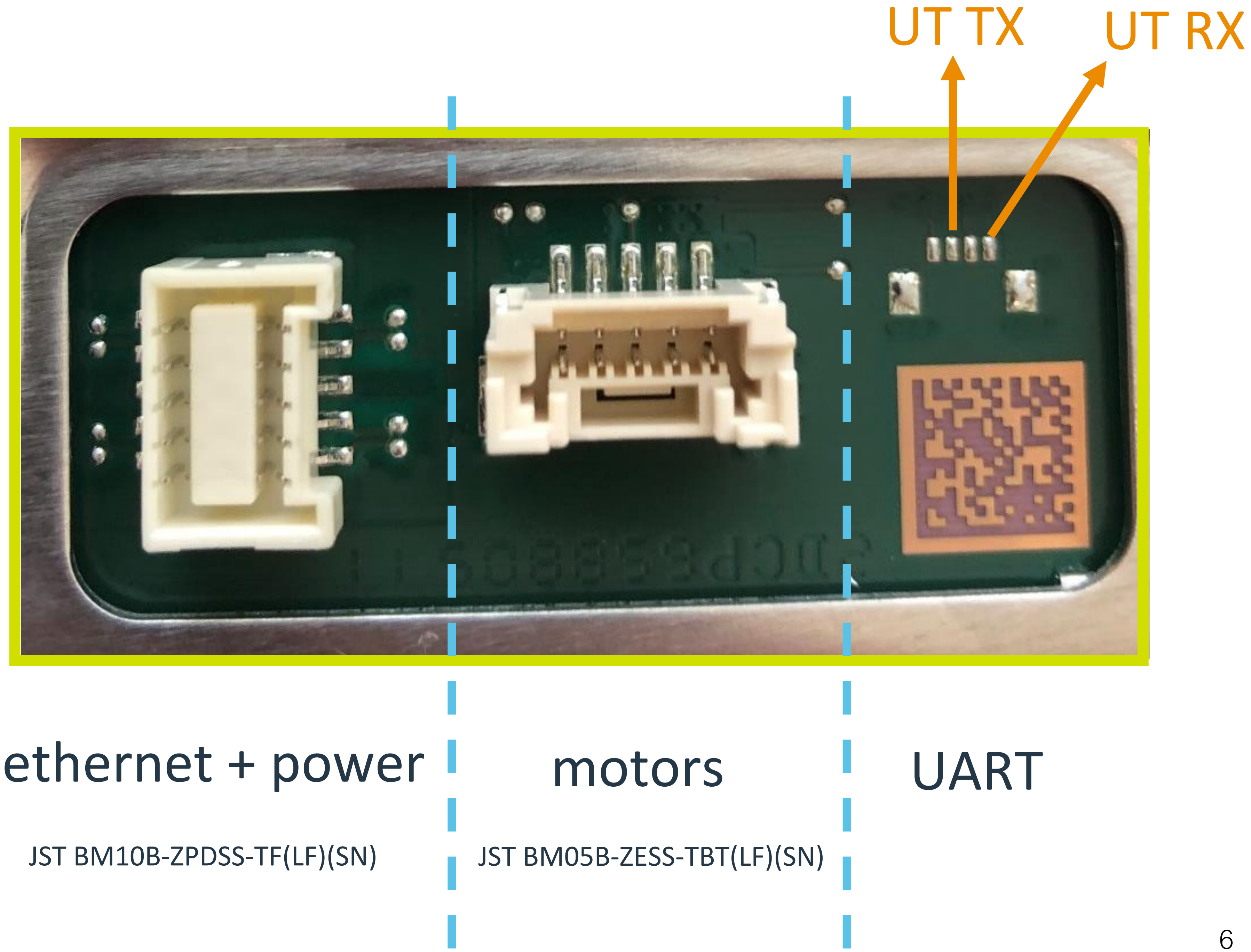
Transceiver

- External phased array
- transceiver_rev2p0/5
- transceiver_rev2p0/5_cut4

This talk (but attack should apply to all UT hardware)

Accessible connectors on V2*





ethernet + power

JST BM10B-ZPDSS-TF(LF)(SN)

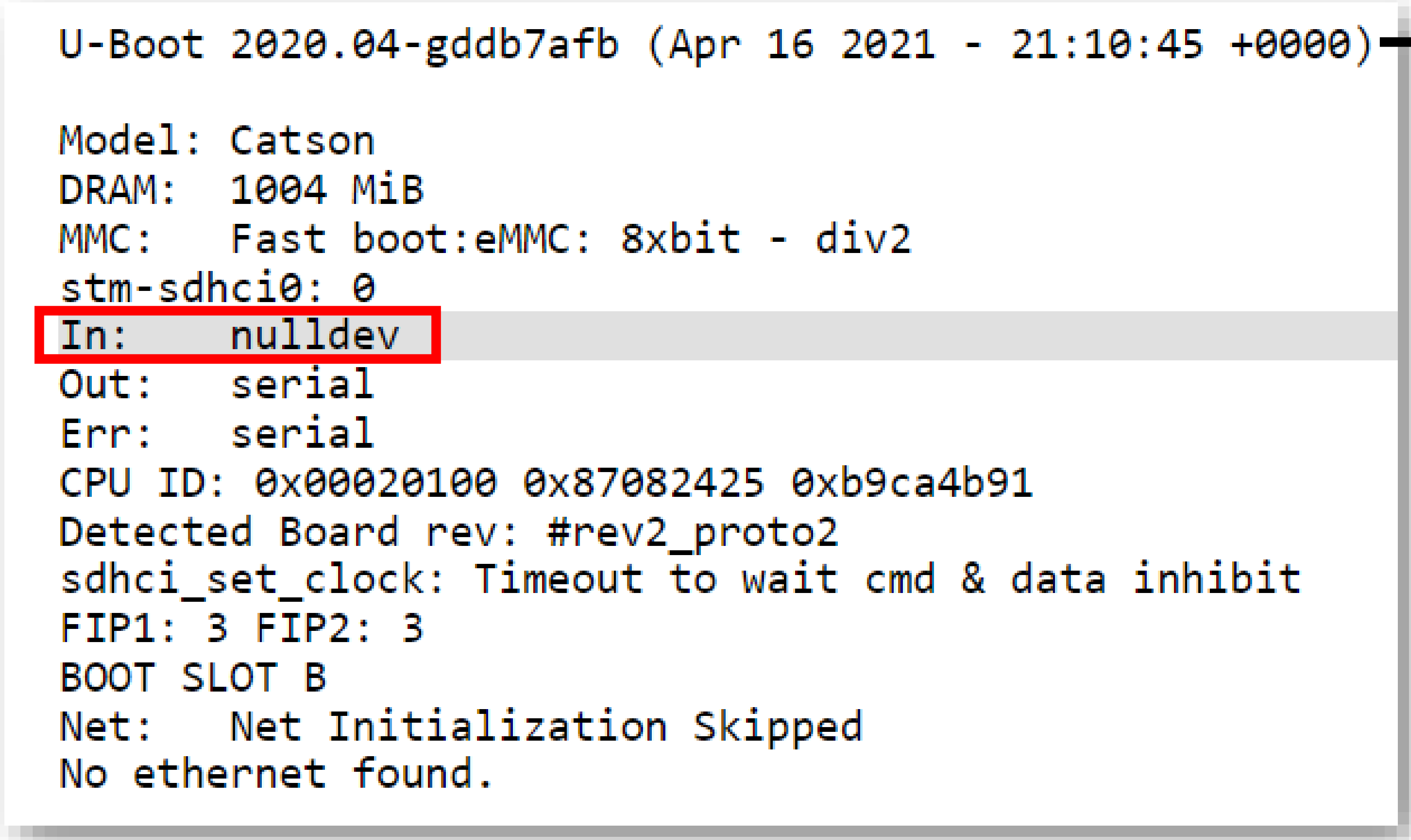
motors

JST BM05B-ZESS-TBT(LF)(SN)

UART

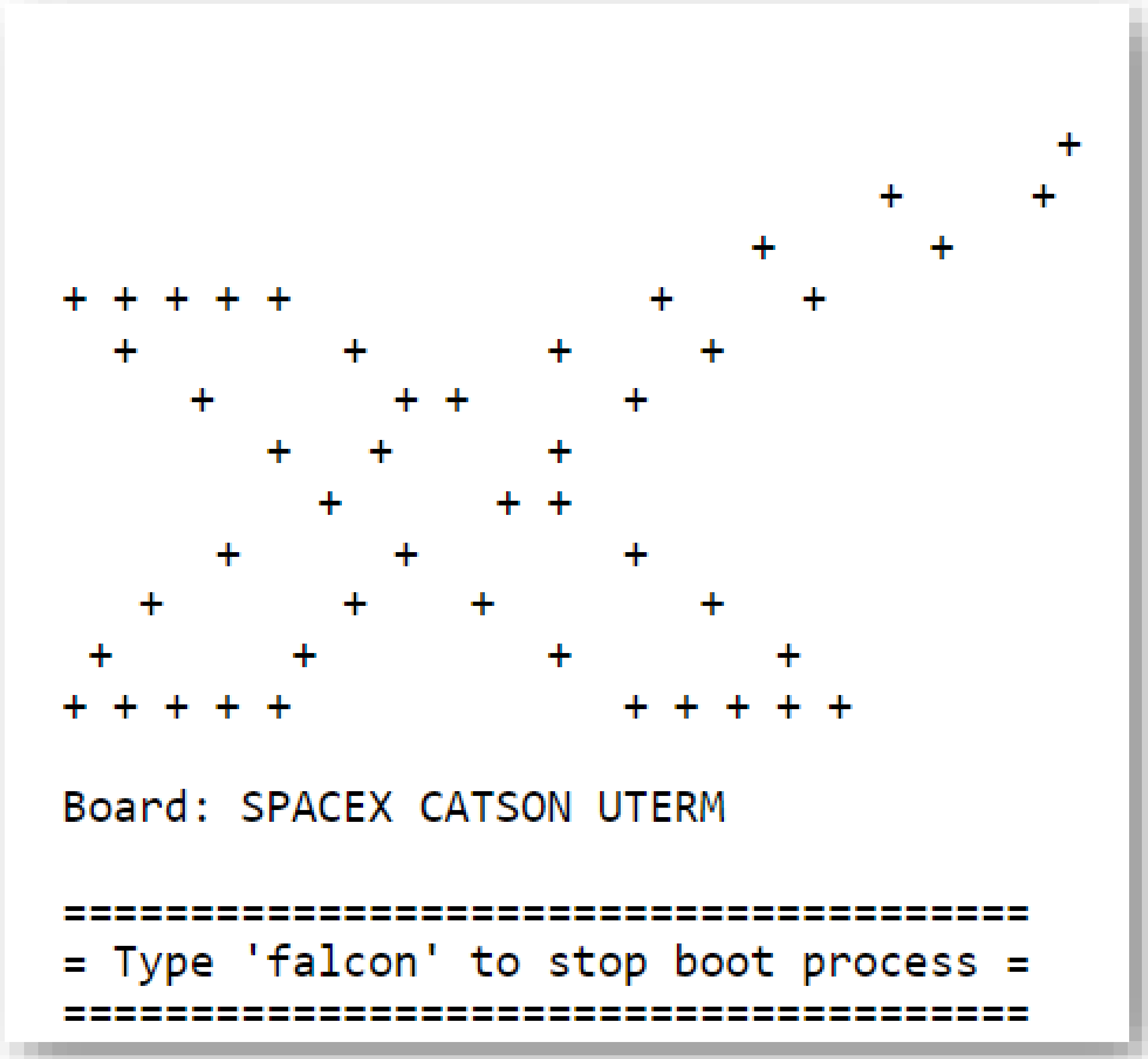
*V1 hardware had an extra connector, V3 does not have easily accessible connectors

UART – U-Boot



```
U-Boot 2020.04-gddb7afb (Apr 16 2021 - 21:10:45 +0000)

Model: Catson
DRAM:  1004 MiB
MMC:   Fast boot:eMMC: 8xbit - div2
stm-sdhci0: 0
In:    nulldev
Out:   serial
Err:   serial
CPU ID: 0x00020100 0x87082425 0xb9ca4b91
Detected Board rev: #rev2_proto2
sdhci_set_clock: Timeout to wait cmd & data inhibit
FIP1: 3 FIP2: 3
BOOT SLOT B
Net:   Net Initialization Skipped
No ethernet found.
```

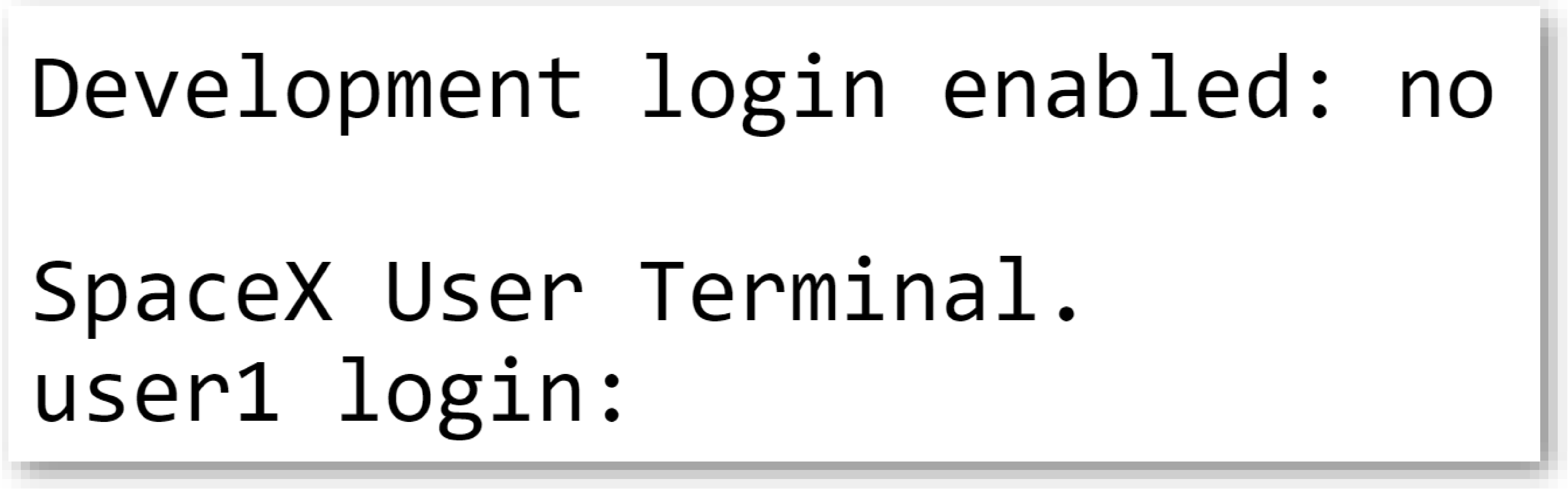
(Newer firmware no longer uses this version)



```
                                       +
                                +     +
                           +      +
+ + + + +              +     +
  +        +       +     +
     +       + +      +
        +   +      +
          +      + +
      +      +        +
   +       +    +        +
 +       +         +        +
+ + + + +             + + + + +

Board: SPACEX CATSON UTERM

======================================
= Type 'falcon' to stop boot process =
======================================
```

U-Boot does not accept serial input
(on non-development/fused hardware)

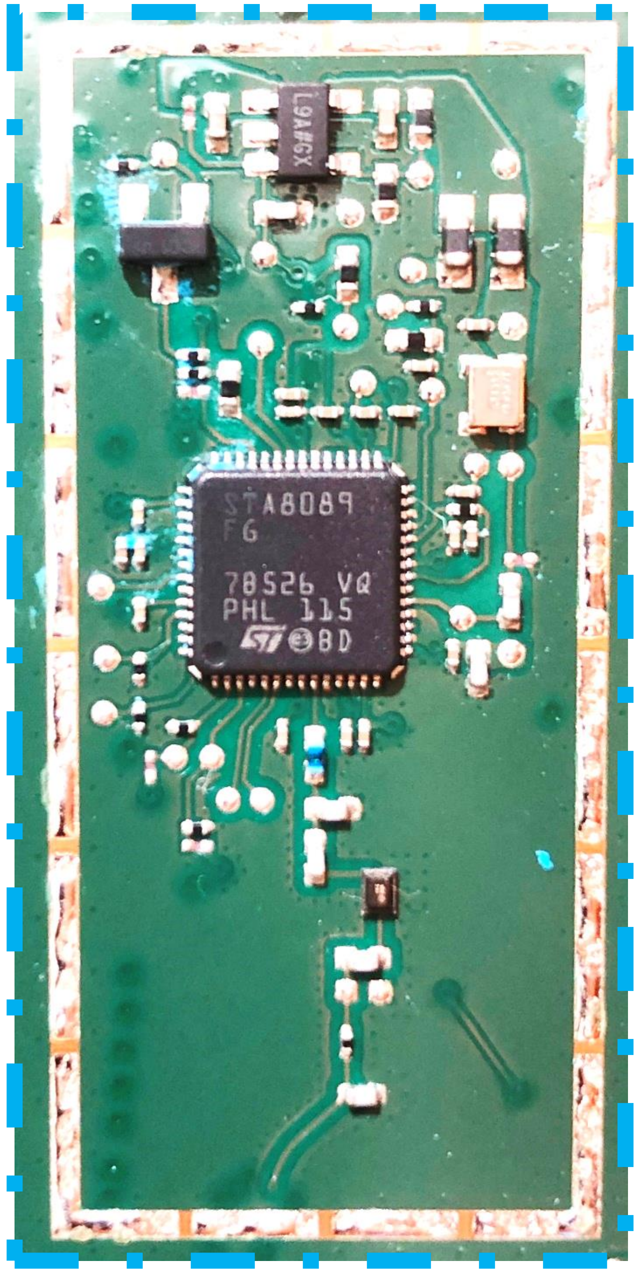
UART – Login Prompt



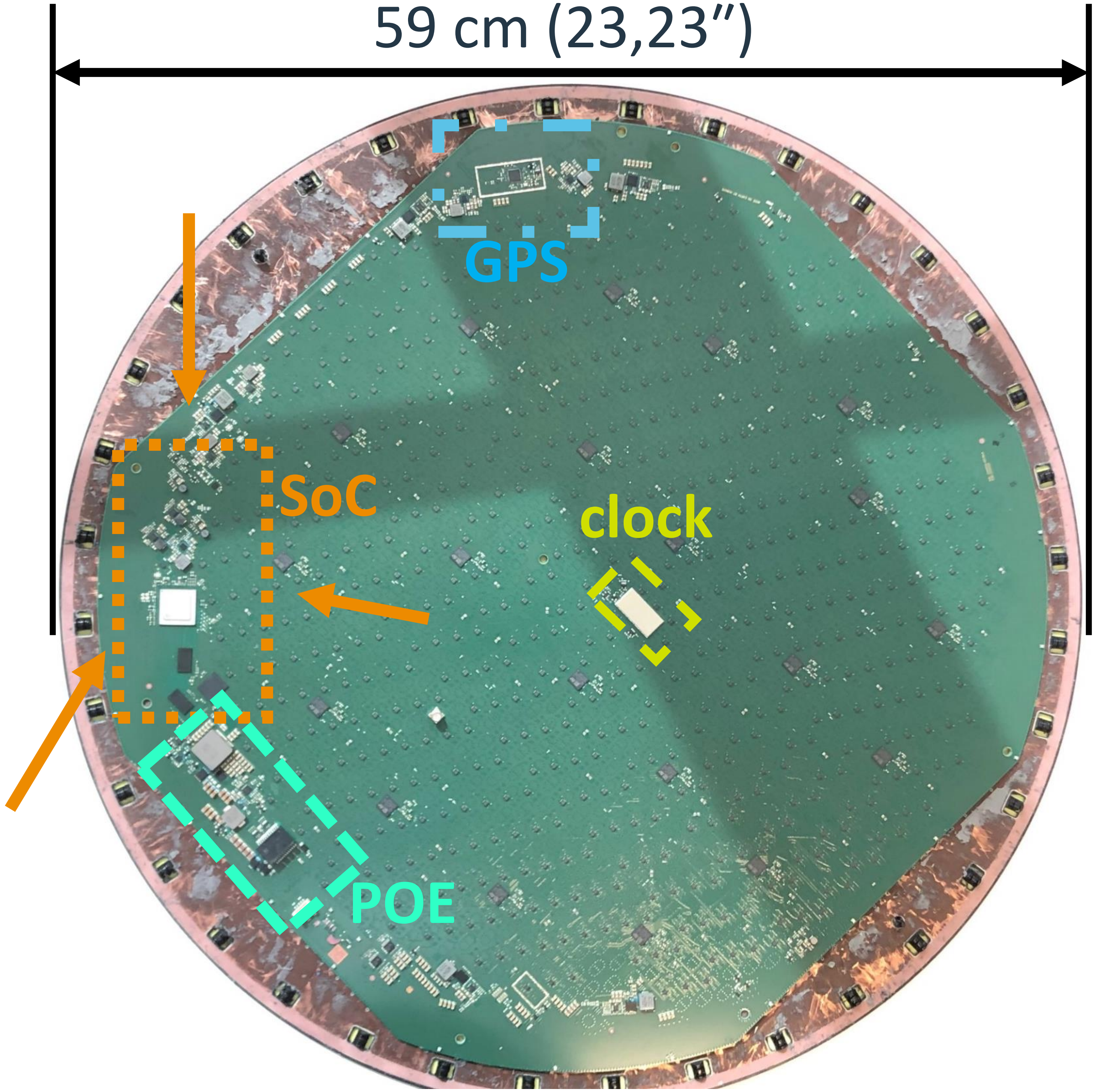
```
Development login enabled: no

SpaceX User Terminal.
user1 login:
```

PCB overview

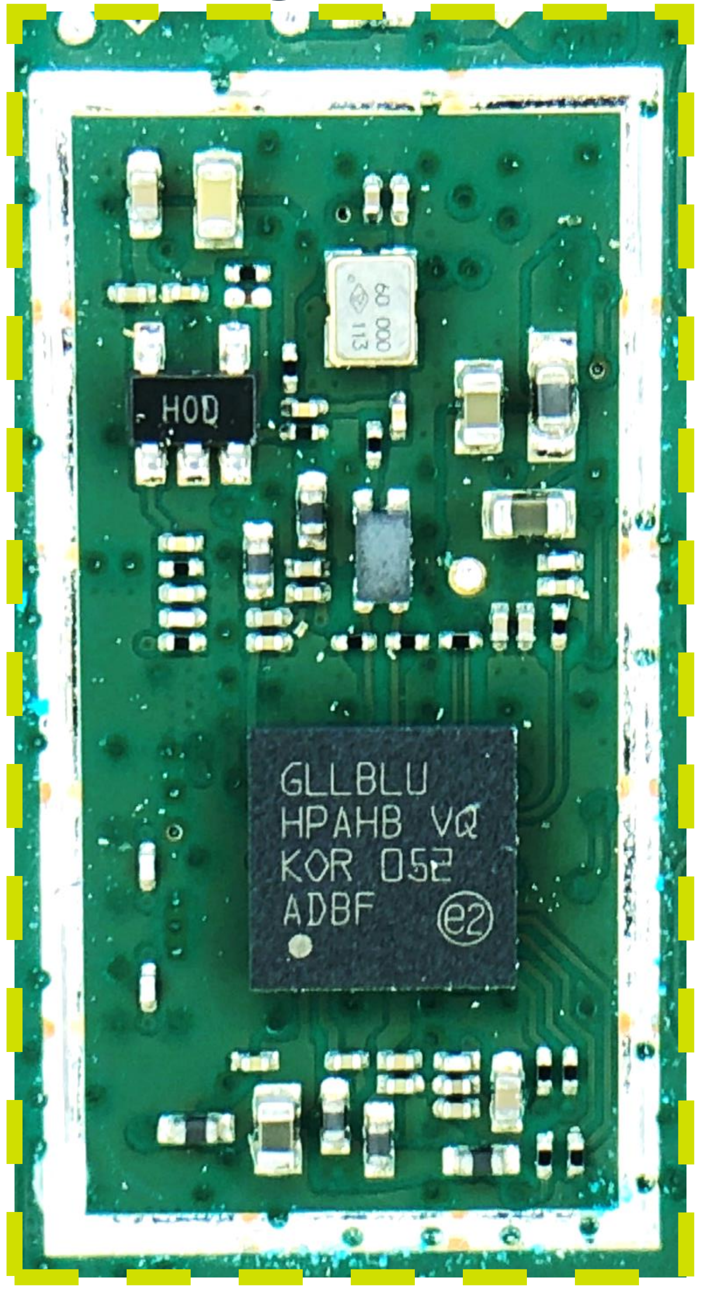
GPS receiver



STM STA8089

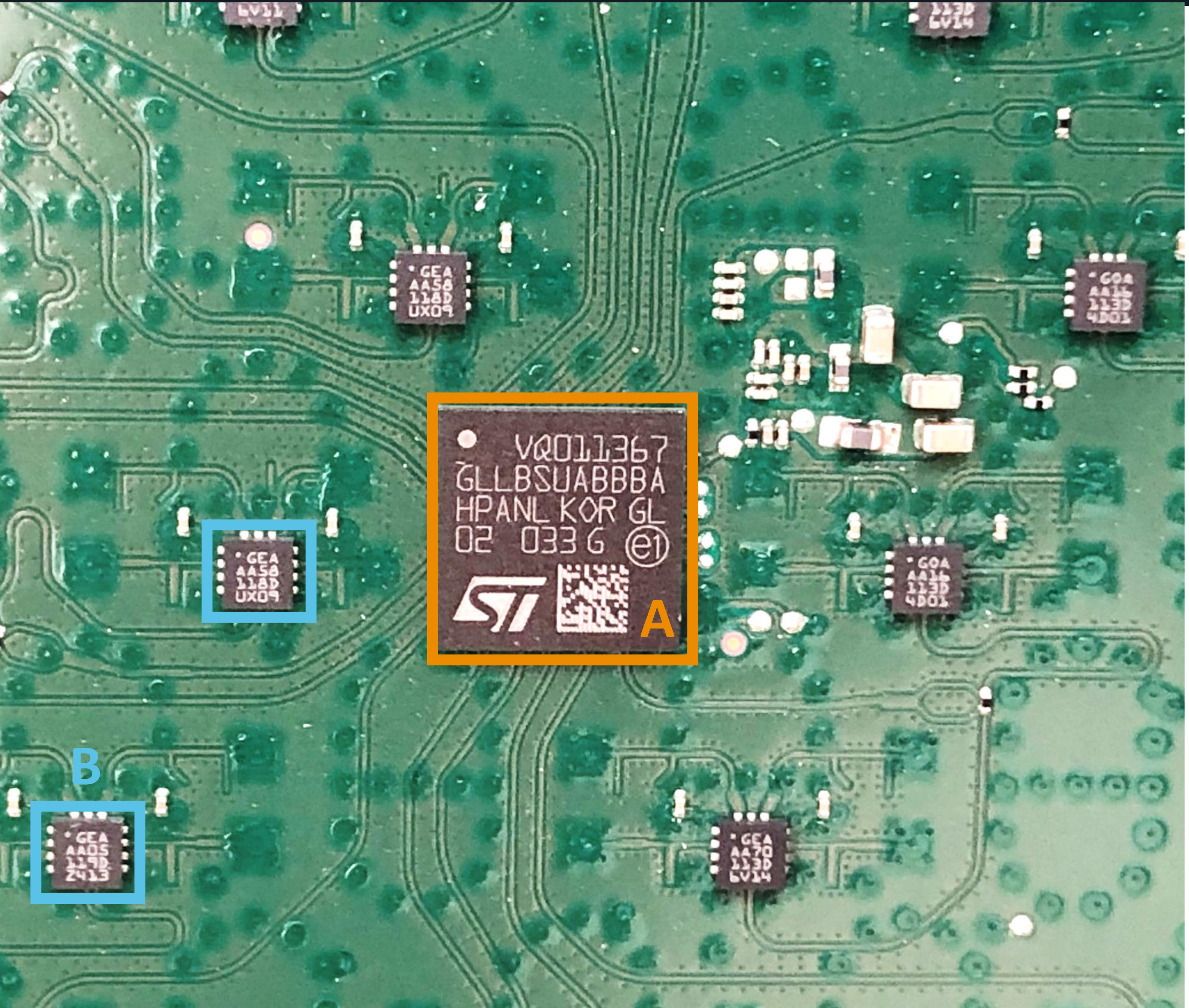


Clock generation



GLLBLU

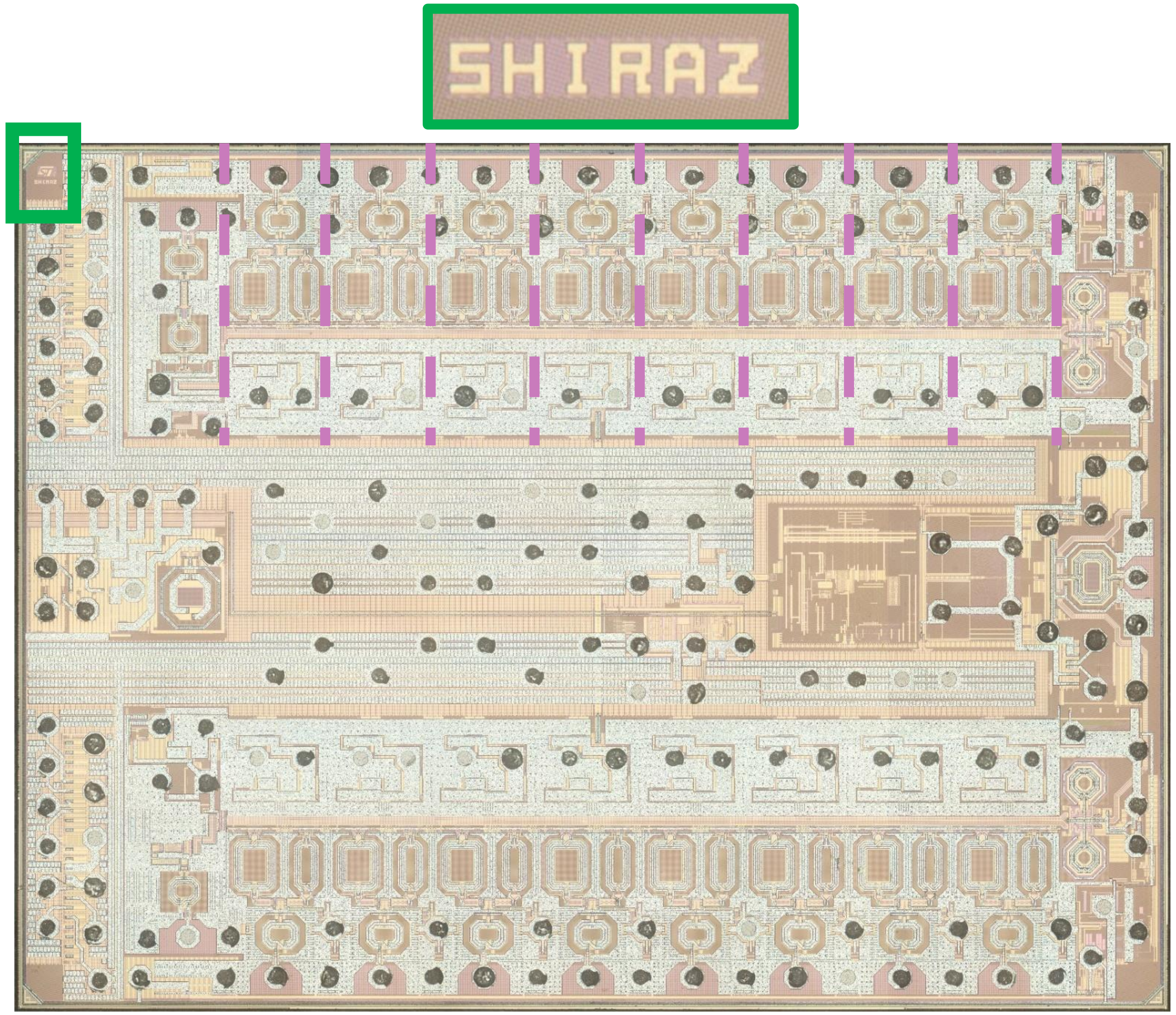
RF Components



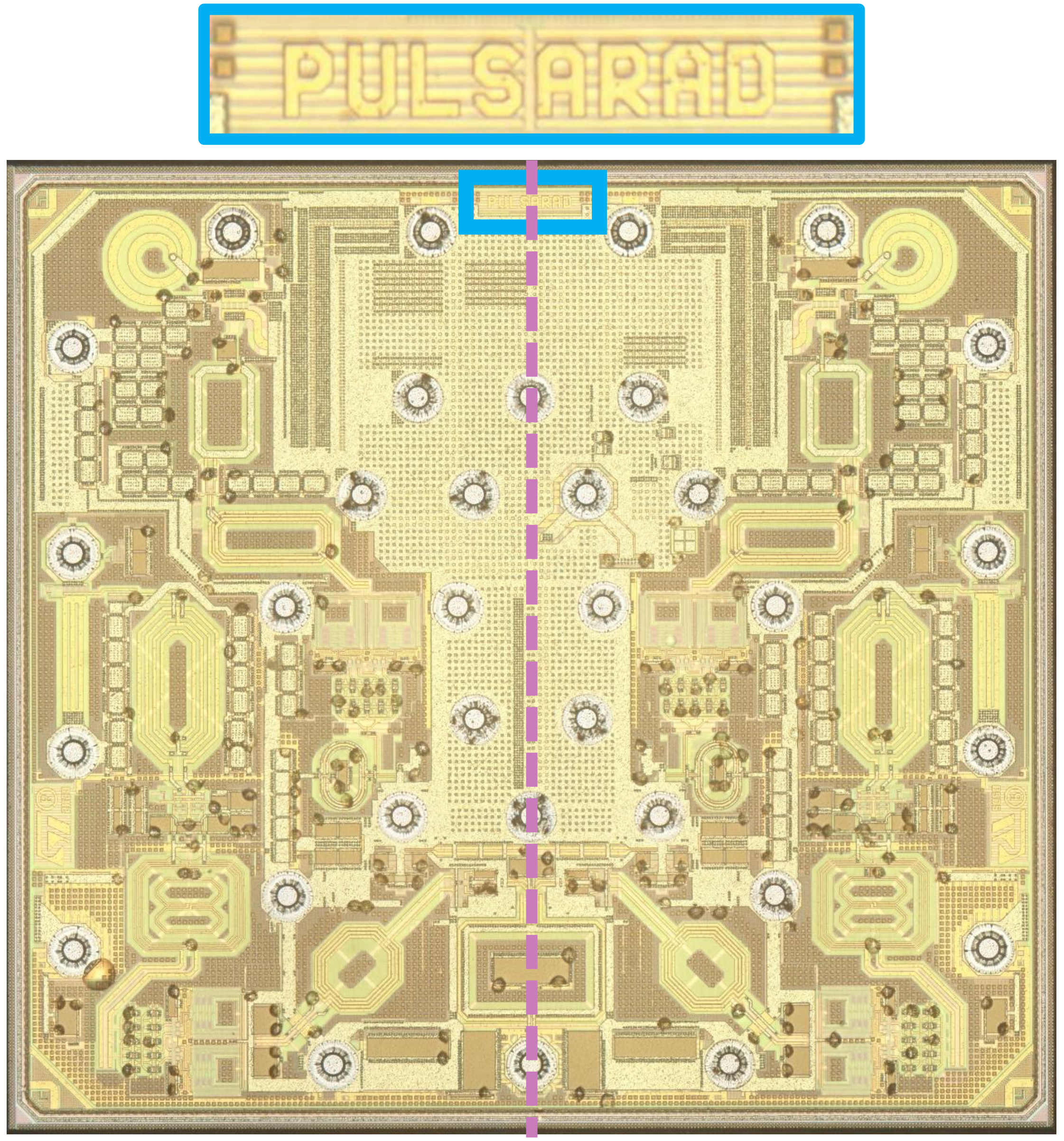
- (A) Digital BeamFormer (DBF)
  - STM GLLBSUABBBA
  - Codename: **SHIRAZ**
  - **NEW: BAMBOO**
- (B) Front-End Module (FEM)
  - Codename: **PULSAR(AD)**

- V2 hardware and up:
  - 1 DBF → 16 FEMs

Siliconpr0n



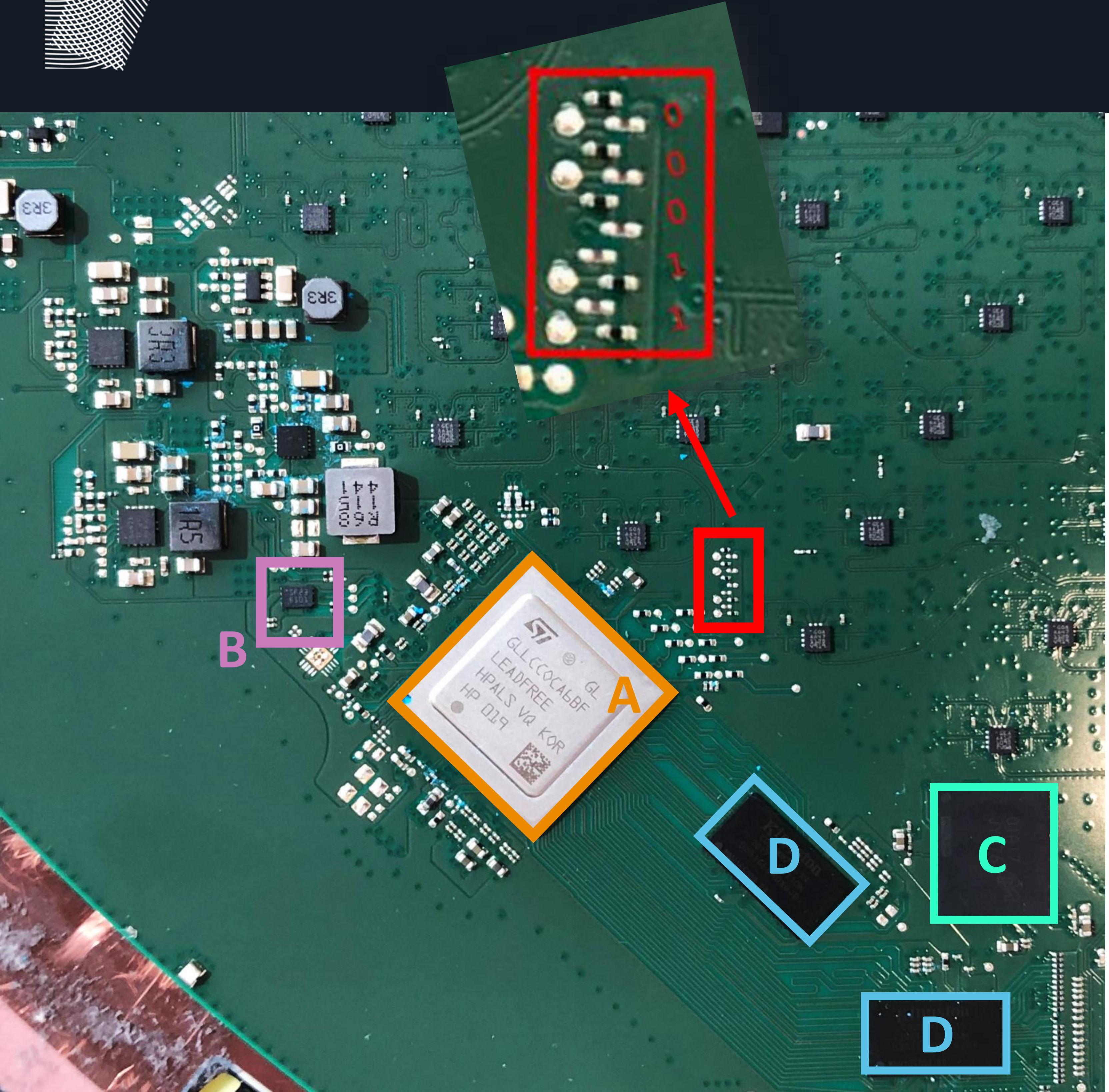
siliconpr0n.org/archive/doku.php?id=mcmaster:spacex:gllbsuabbba-shiraz



id=mcmaster:spacex:gea-aa12-109d-tg02-pulsarad

Thanks to John McMaster!

@johndmcmaster



- (A) System-on-Chip
  - Custom quad-core ARM Cortex-A53
  - ST Microelectronics
    - GLLCCO**C**A6BF (cut 3?)
    - GLLCCO**D**A6BF (cut 4?)
  - Codename: **CATSON**

- (B) Secure Element
  - STM STSAFE-A110

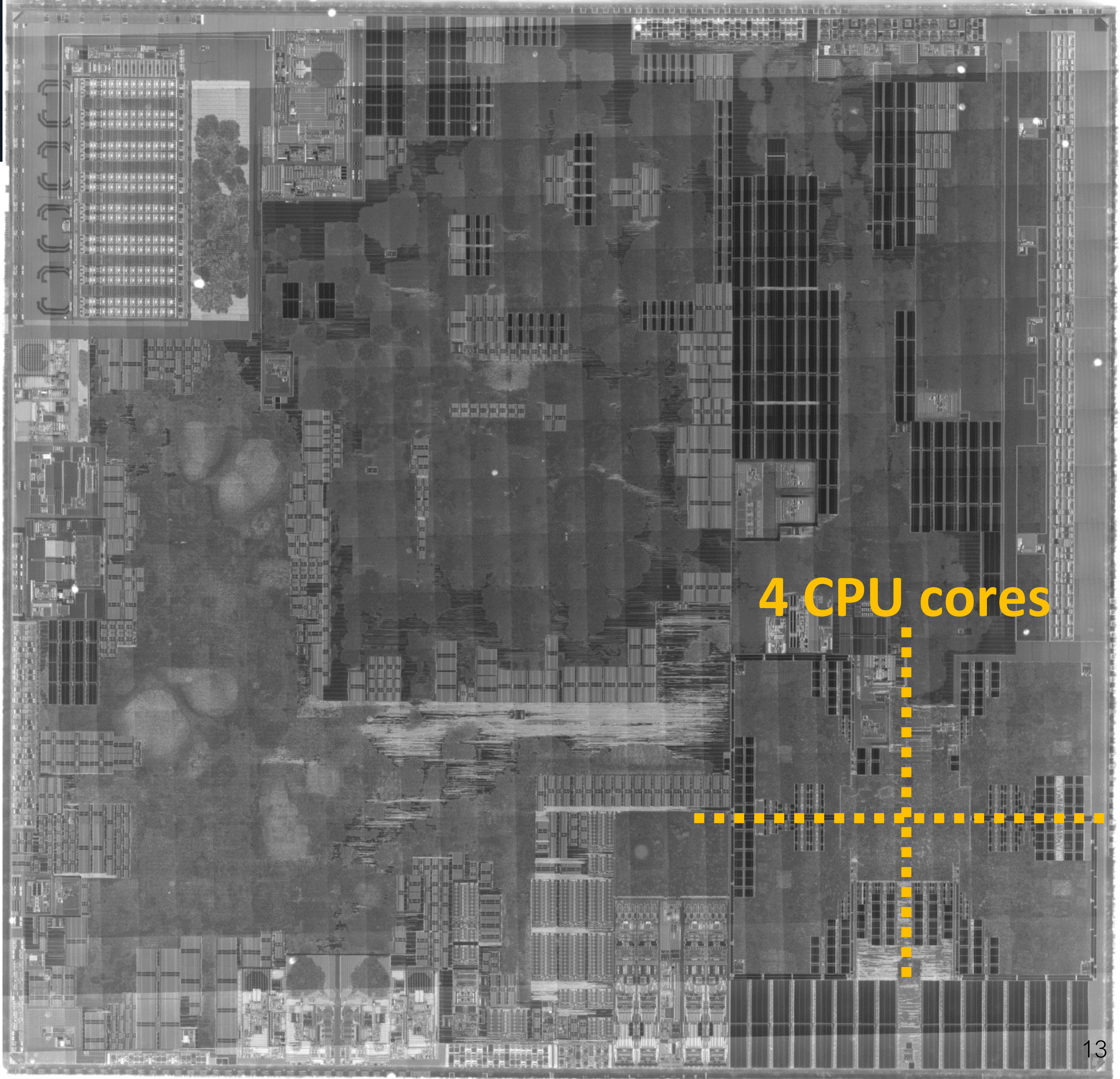
- (C) 4GB eMMC
- (D) 2 x 4Gbit DDR3

SoC

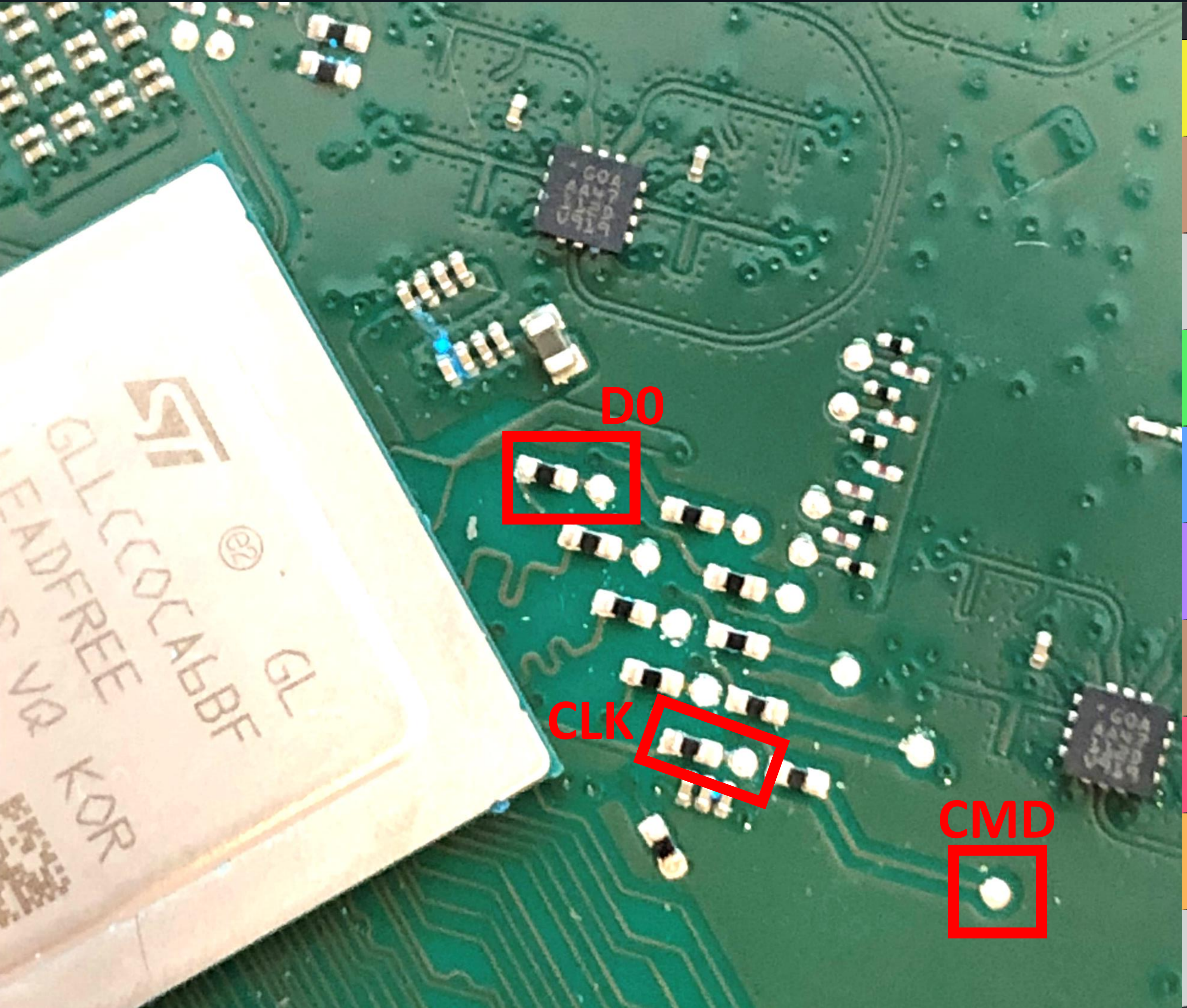
- through substrate image
  - GLLCCOCA6BF (cut 3?)
  - Thorlabs NIR camera
  - Mitutoyo NIR objective 50x

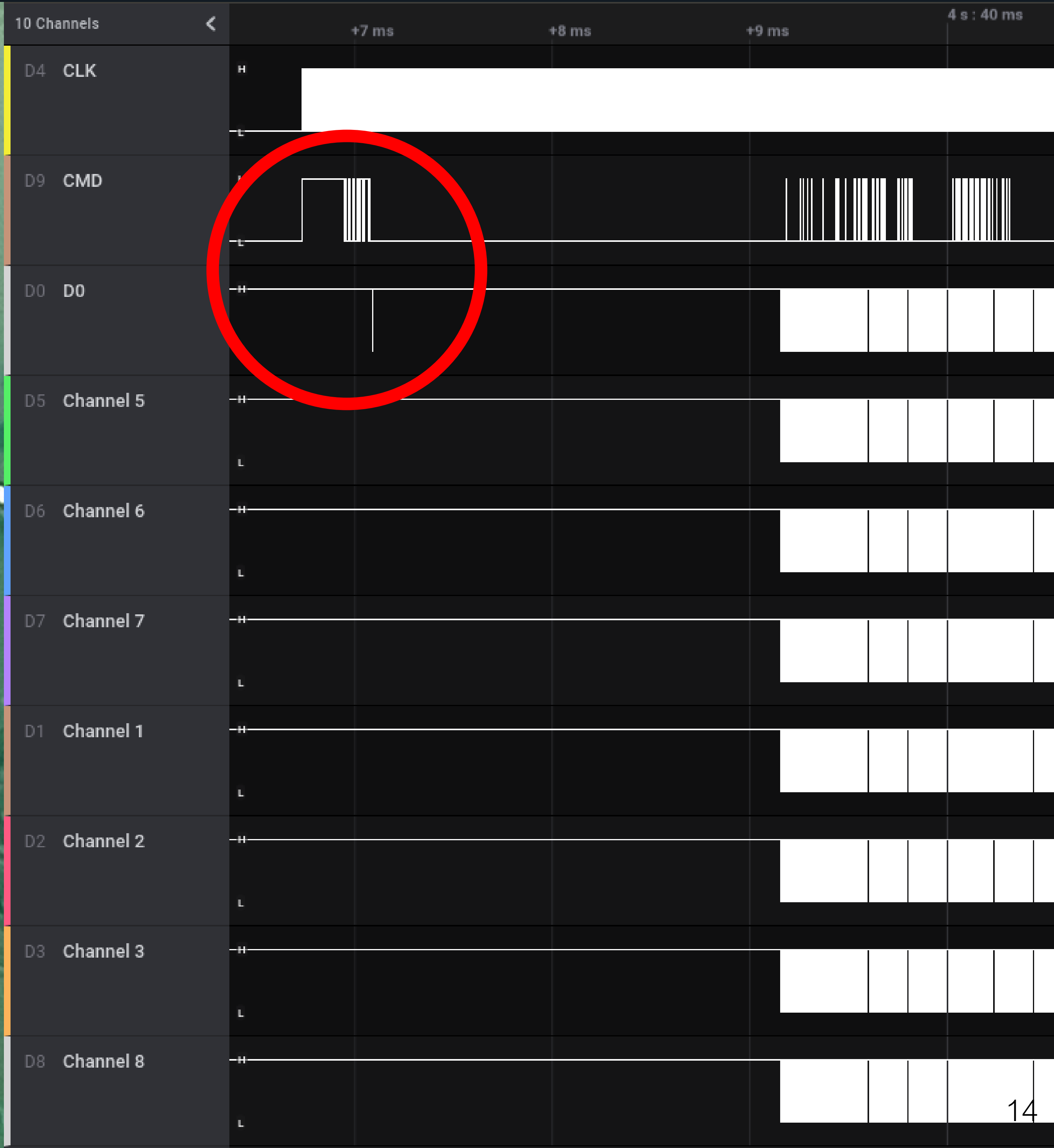
- Can help narrow down interesting locations for some physical attacks

- Full resolution version is available on siliconpr0n.org!

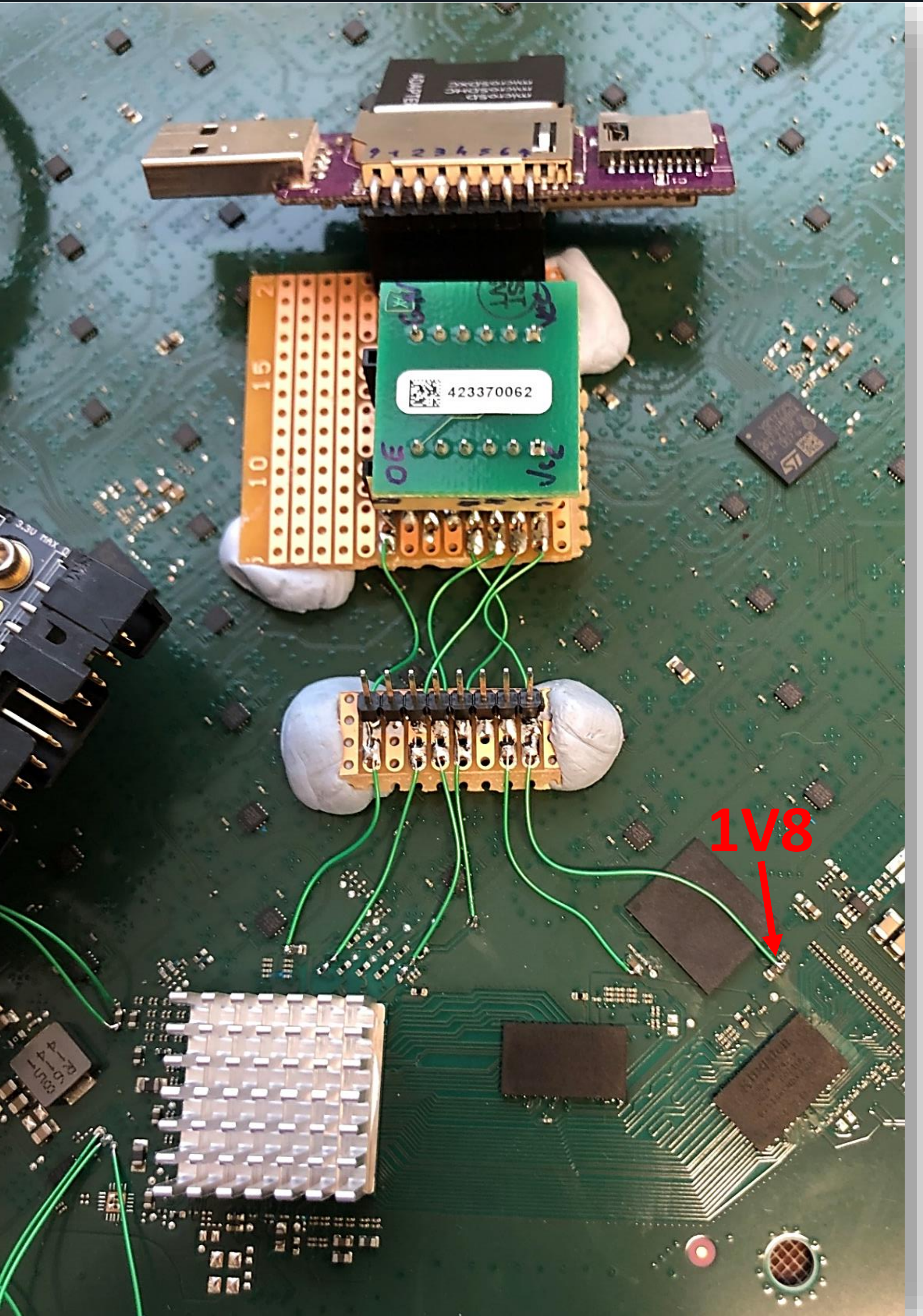


Identifying eMMC test points





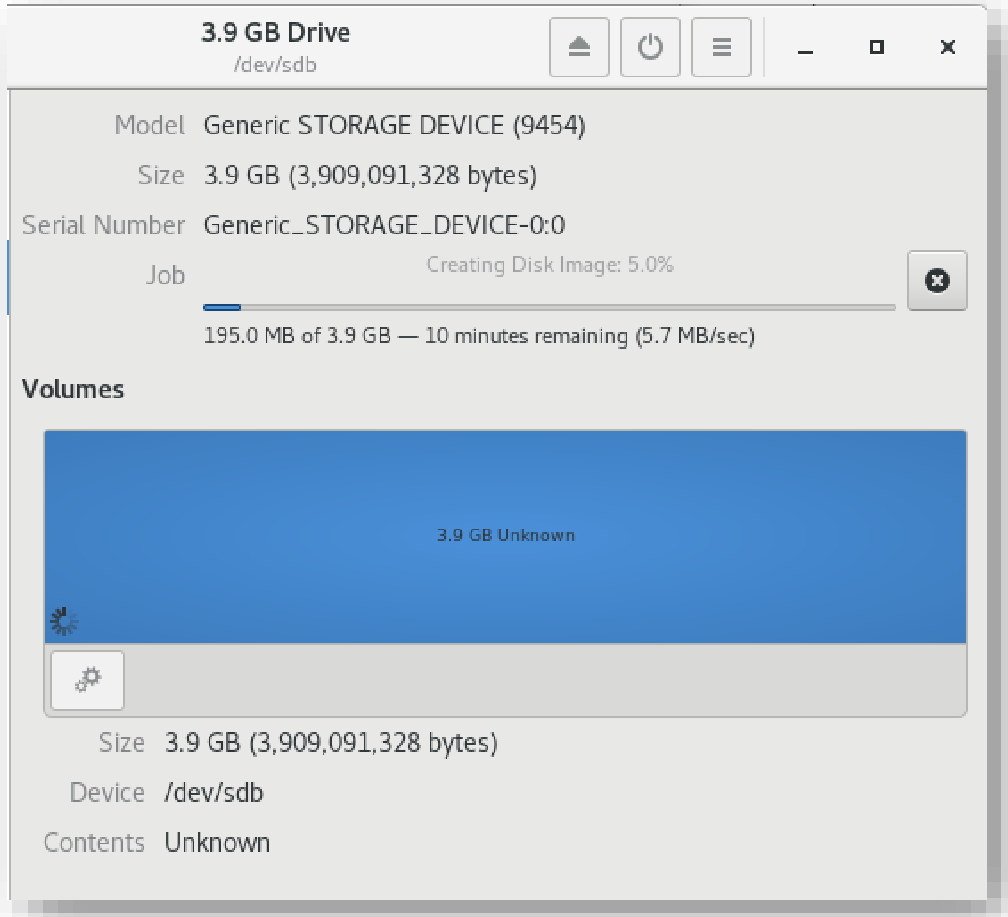
Reading eMMC in-circuit



What I did

SD card reader

TXS0202EVM
Level shifter



3.9 GB Drive
/dev/sdb

Model Generic STORAGE DEVICE (9454)
Size 3.9 GB (3,909,091,328 bytes)
Serial Number Generic_STORAGE_DEVICE-0:0
Job Creating Disk Image: 5.0%
195.0 MB of 3.9 GB — 10 minutes remaining (5.7 MB/sec)

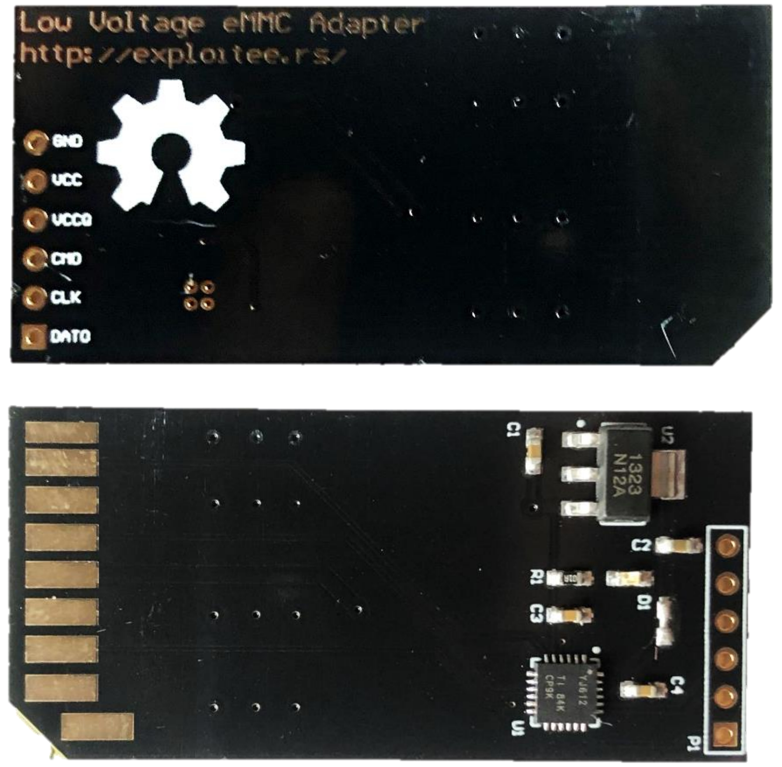
Volumes

3.9 GB Unknown

Size 3.9 GB (3,909,091,328 bytes)
Device /dev/sdb
Contents Unknown

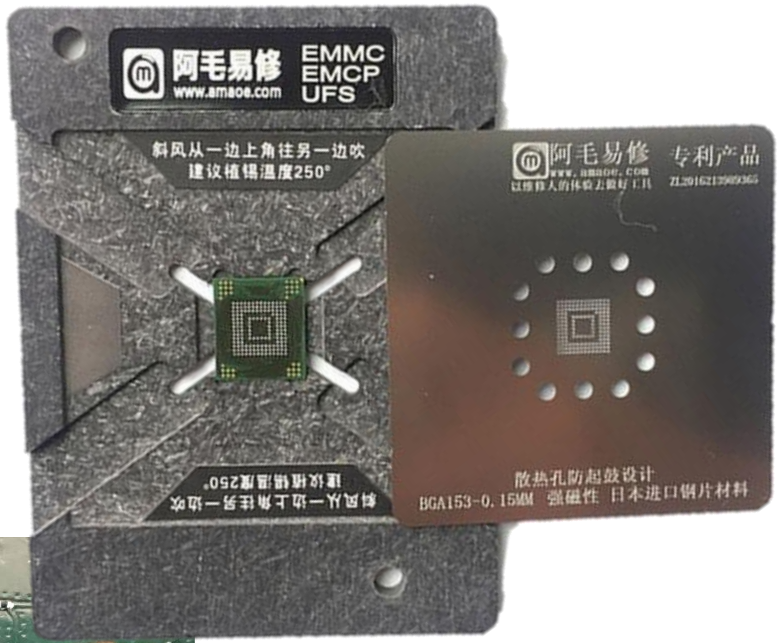
What I recommend

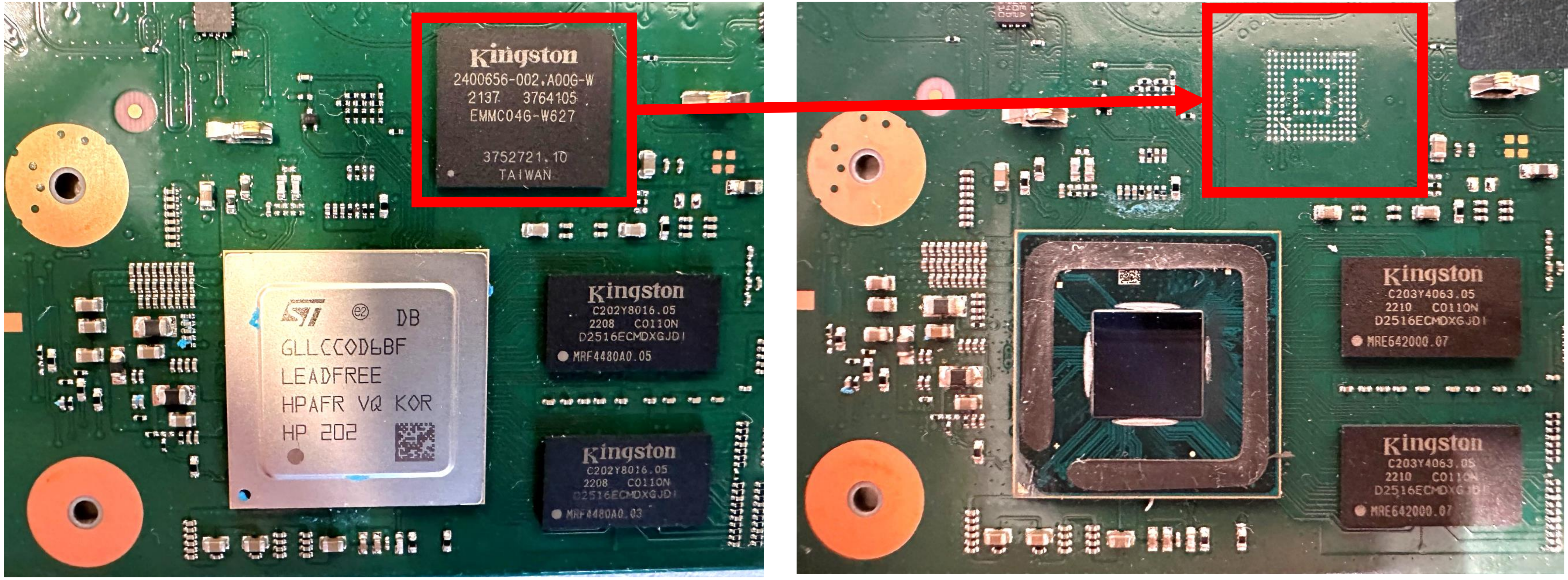
Low Voltage eMMC Adapter
by
exploitee.rs

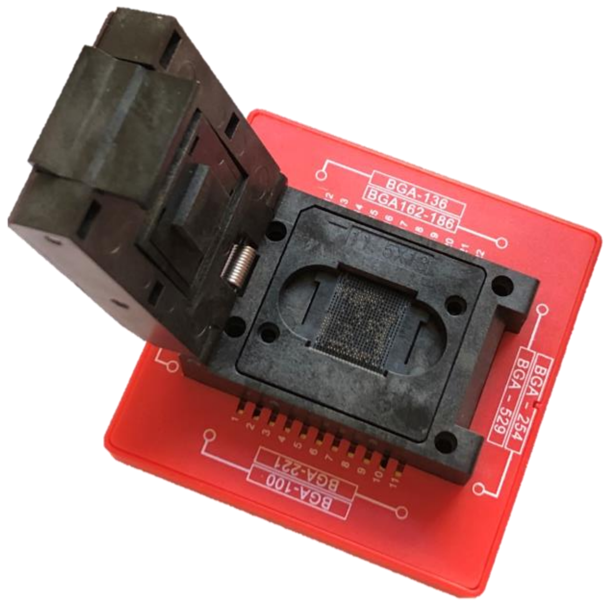


eMMC chip-off

- Hot air to remove the eMMC from the board
- “eMMC reballing jig” and solder paste
- eMMC adapter

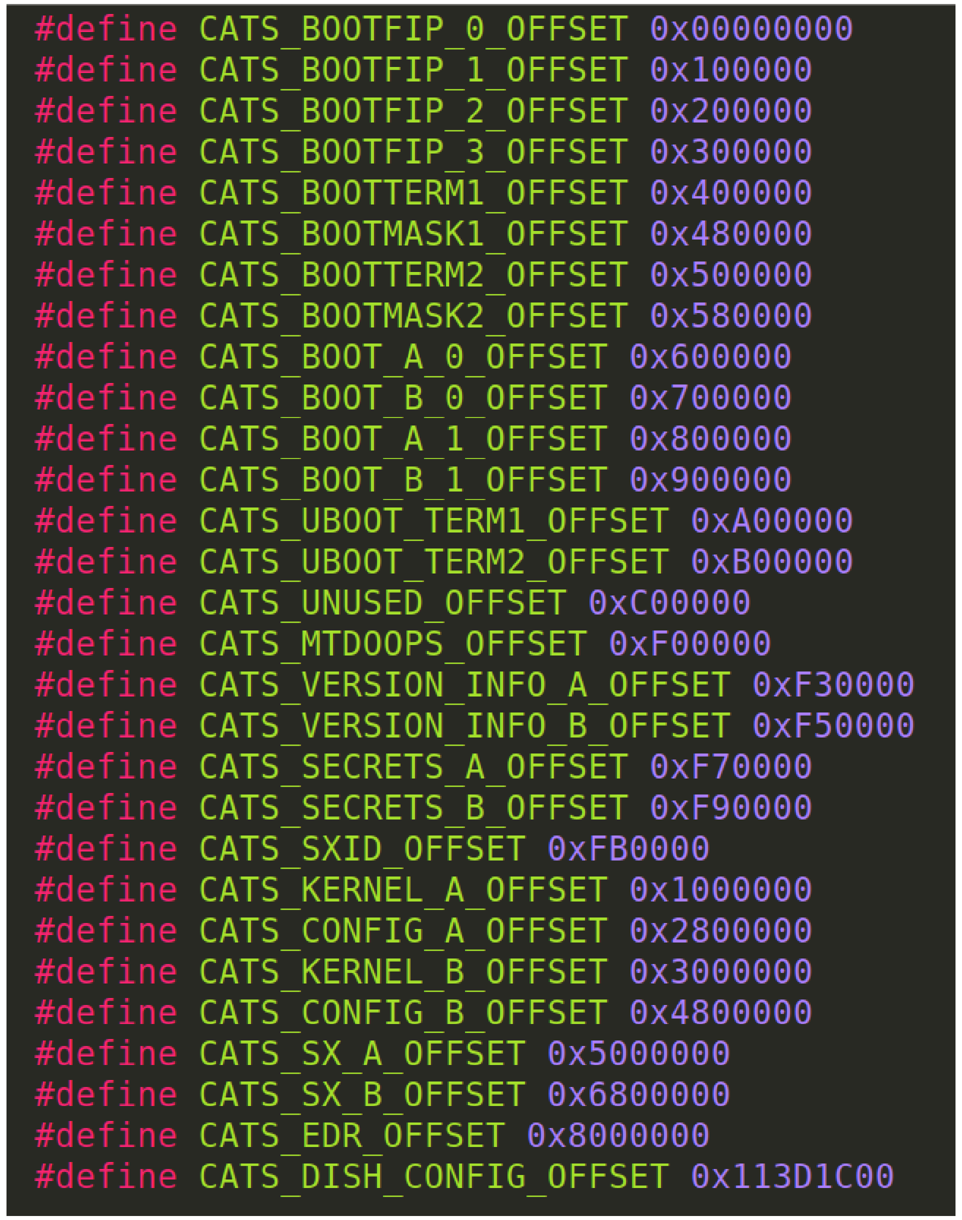





MOORC eMMC socket

Extracting the eMMC dump

- Split the dump into:
  - TF-A Bootstages: Firmware Image Packages
    - unpack with TF-A fiptool
  - Flattened uImage Tree (FIT, custom SXECC format)
    - unpack with U-Boot dumpimage
  - SpaceX Runtime (dm-verity)
  - SpaceX Calibration (dm-verity)
  - SpaceX EDR (LUKS)
  - SpaceX dish config (LUKS)
    - LUKS keys are stored in eFuses



```
#define CATS_BOOTFIP_0_OFFSET 0x00000000
#define CATS_BOOTFIP_1_OFFSET 0x100000
#define CATS_BOOTFIP_2_OFFSET 0x200000
#define CATS_BOOTFIP_3_OFFSET 0x300000
#define CATS_BOOTTERM1_OFFSET 0x400000
#define CATS_BOOTMASK1_OFFSET 0x480000
#define CATS_BOOTTERM2_OFFSET 0x500000
#define CATS_BOOTMASK2_OFFSET 0x580000
#define CATS_BOOT_A_0_OFFSET 0x600000
#define CATS_BOOT_B_0_OFFSET 0x700000
#define CATS_BOOT_A_1_OFFSET 0x800000
#define CATS_BOOT_B_1_OFFSET 0x900000
#define CATS_UBOOT_TERM1_OFFSET 0xA00000
#define CATS_UBOOT_TERM2_OFFSET 0xB00000
#define CATS_UNUSED_OFFSET 0xC00000
#define CATS_MTDOOPS_OFFSET 0xF00000
#define CATS_VERSION_INFO_A_OFFSET 0xF30000
#define CATS_VERSION_INFO_B_OFFSET 0xF50000
#define CATS_SECRETS_A_OFFSET 0xF70000
#define CATS_SECRETS_B_OFFSET 0xF90000
#define CATS_SXID_OFFSET 0xFB0000
#define CATS_KERNEL_A_OFFSET 0x1000000
#define CATS_CONFIG_A_OFFSET 0x2800000
#define CATS_KERNEL_B_OFFSET 0x3000000
#define CATS_CONFIG_B_OFFSET 0x4800000
#define CATS_SX_A_OFFSET 0x5000000
#define CATS_SX_B_OFFSET 0x6800000
#define CATS_EDR_OFFSET 0x8000000
#define CATS_DISH_CONFIG_OFFSET 0x113D1C00
```

U-Boot GPL sources: spacex_catson_boot.h

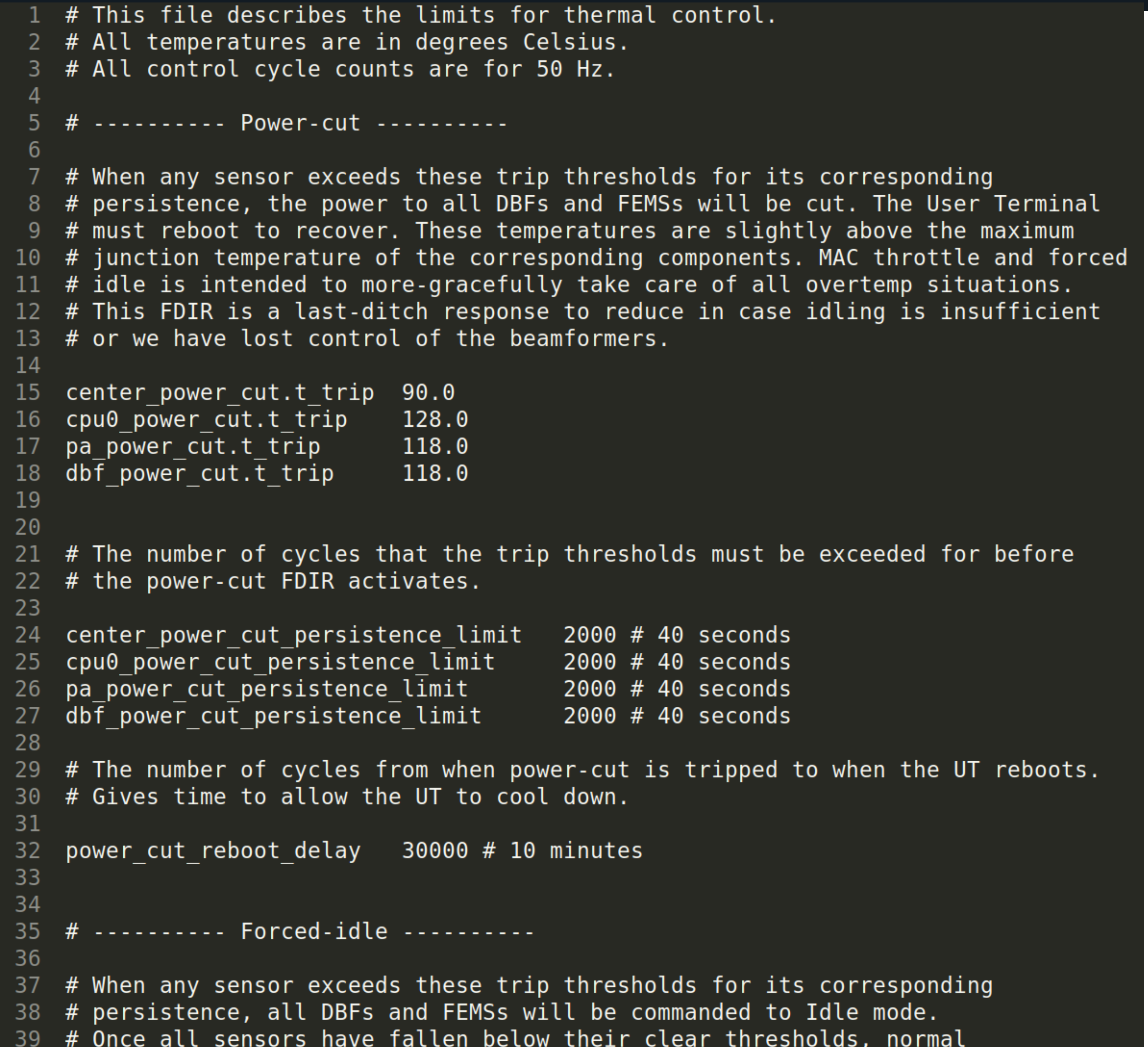
Unpacking the FIT

- Binwalk and dumpimage failed to extract the FIT

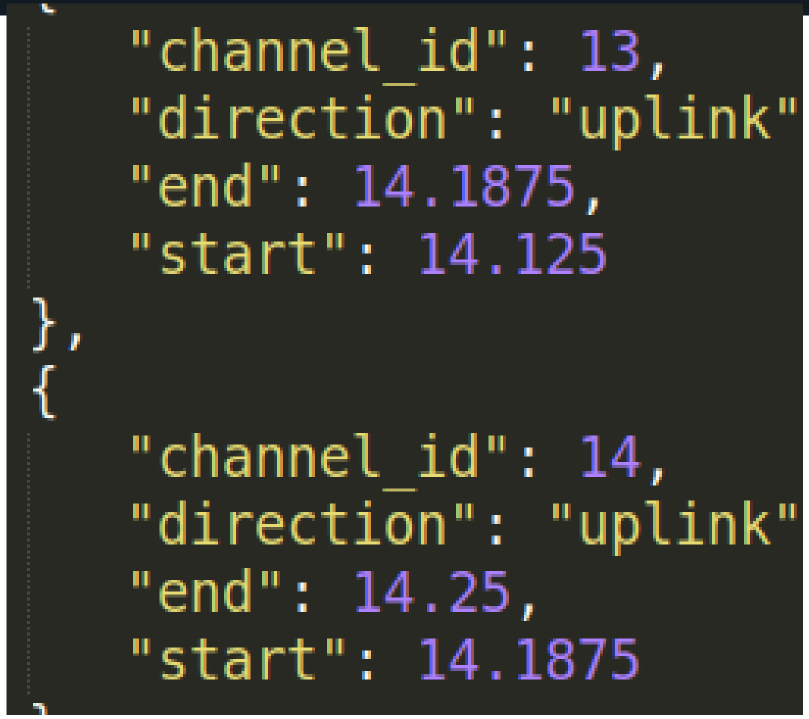
- Custom format (SXECC magic)
  - U-Boot GPL sources: https://github.com/SpaceExplorationTechnologies
- Each 255-byte block of data contains 32-bytes of error correcting codes
- Full format explained here:
    - esat.kuleuven.be/cosic/blog/dumping-and-extracting-the-spacex-starlink-user-terminal-firmware

- After stripping the ECC once you can use the included unecc binary:
  chroot . ./qemu-aarch64-static ./usr/bin/unecc -c < infile > outfile

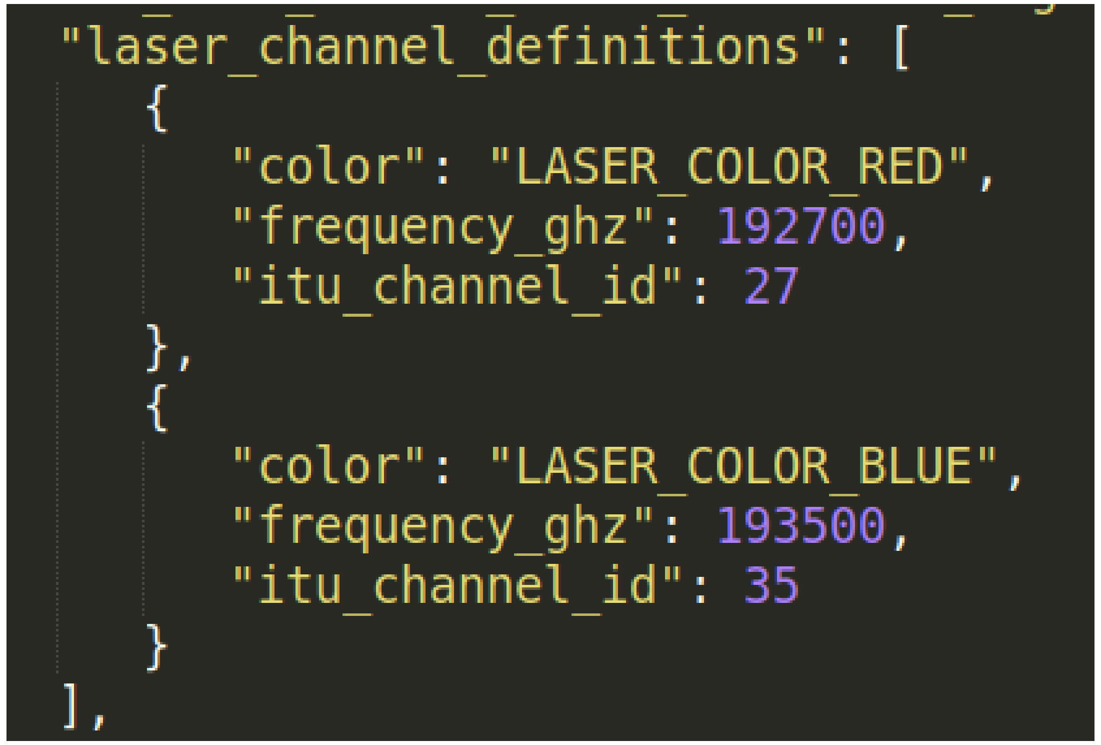
Temperature and RF channels



```
# This file describes the limits for thermal control.
# All temperatures are in degrees Celsius.
# All control cycle counts are for 50 Hz.

# ---------- Power-cut ----------

# When any sensor exceeds these trip thresholds for its corresponding
# persistence, the power to all DBFs and FEMSs will be cut. The User Terminal
# must reboot to recover. These temperatures are slightly above the maximum
# junction temperature of the corresponding components. MAC throttle and forced
# idle is intended to more-gracefully take care of all overtemp situations.
# This FDIR is a last-ditch response to reduce in case idling is insufficient
# or we have lost control of the beamformers.

center_power_cut.t_trip  90.0
cpu0_power_cut.t_trip    128.0
pa_power_cut.t_trip      118.0
dbf_power_cut.t_trip     118.0


# The number of cycles that the trip thresholds must be exceeded for before
# the power-cut FDIR activates.

center_power_cut_persistence_limit   2000 # 40 seconds
cpu0_power_cut_persistence_limit     2000 # 40 seconds
pa_power_cut_persistence_limit       2000 # 40 seconds
dbf_power_cut_persistence_limit      2000 # 40 seconds

# The number of cycles from when power-cut is tripped to when the UT reboots.
# Gives time to allow the UT to cool down.

power_cut_reboot_delay   30000 # 10 minutes


# ---------- Forced-idle ----------

# When any sensor exceeds these trip thresholds for its corresponding
# persistence, all DBFs and FEMSs will be commanded to Idle mode.
# Once all sensors have fallen below their clear thresholds, normal
```

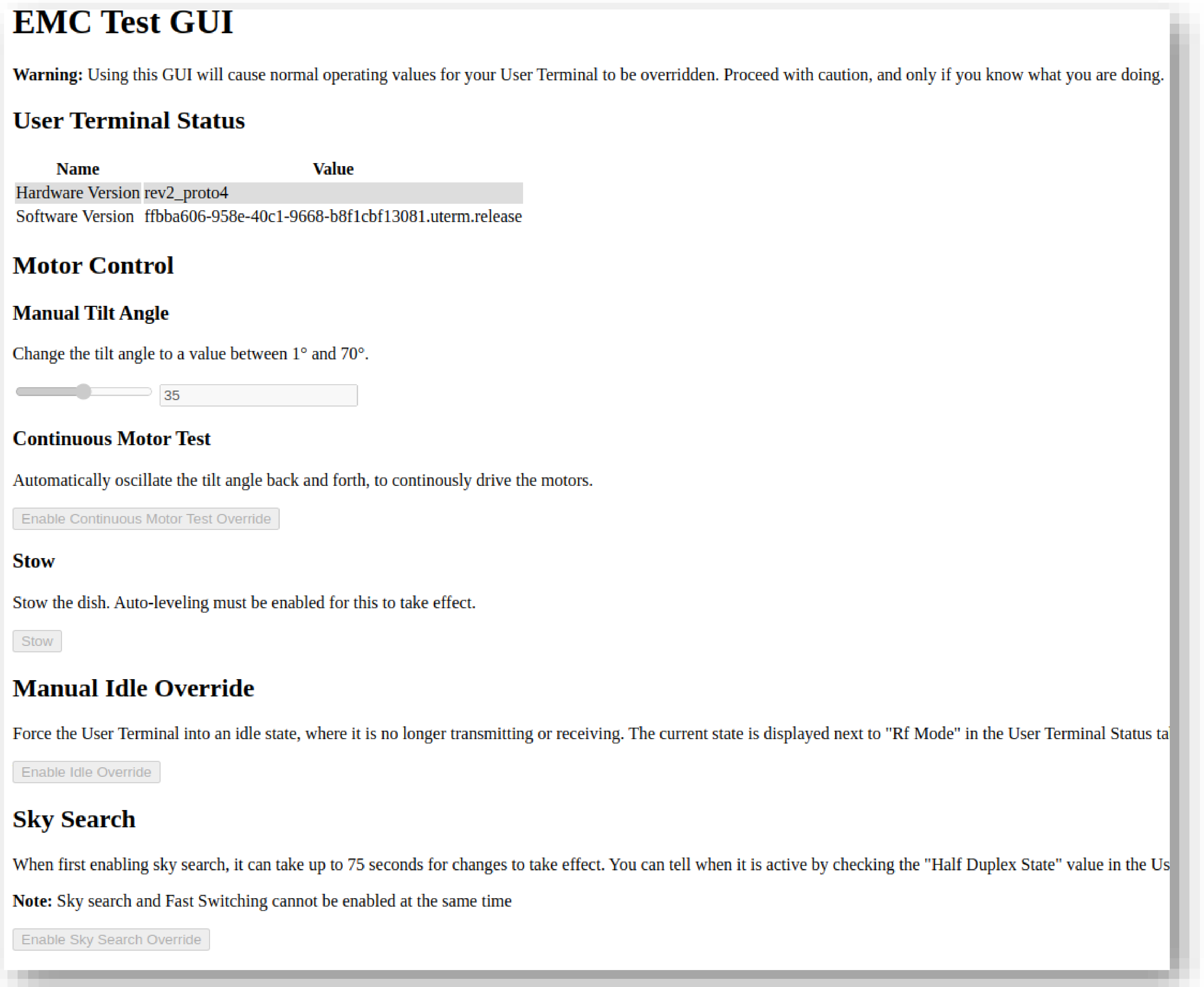


```
    "channel_id": 13,
    "direction": "uplink"
    "end": 14.1875,
    "start": 14.125
  },
  {
    "channel_id": 14,
    "direction": "uplink"
    "end": 14.25,
    "start": 14.1875
```



```
"laser_channel_definitions": [
  {
    "color": "LASER_COLOR_RED",
    "frequency_ghz": 192700,
    "itu_channel_id": 27
  },
  {
    "color": "LASER_COLOR_BLUE",
    "frequency_ghz": 193500,
    "itu_channel_id": 35
  }
],
```

Webpages



EMC Test GUI

Warning: Using this GUI will cause normal operating values for your User Terminal to be overridden. Proceed with caution, and only if you know what you are doing.

User Terminal Status

Name	Value
Hardware Version	rev2_proto4
Software Version	ffbba606-958e-40c1-9668-b8f1cbf13081.uterm.release

Motor Control

Manual Tilt Angle

Change the tilt angle to a value between 1° and 70°.

35

Continuous Motor Test

Automatically oscillate the tilt angle back and forth, to continously drive the motors.

Enable Continuous Motor Test Override

Stow

Stow the dish. Auto-leveling must be enabled for this to take effect.

Stow

Manual Idle Override

Force the User Terminal into an idle state, where it is no longer transmitting or receiving. The current state is displayed next to "Rf Mode" in the User Terminal Status ta

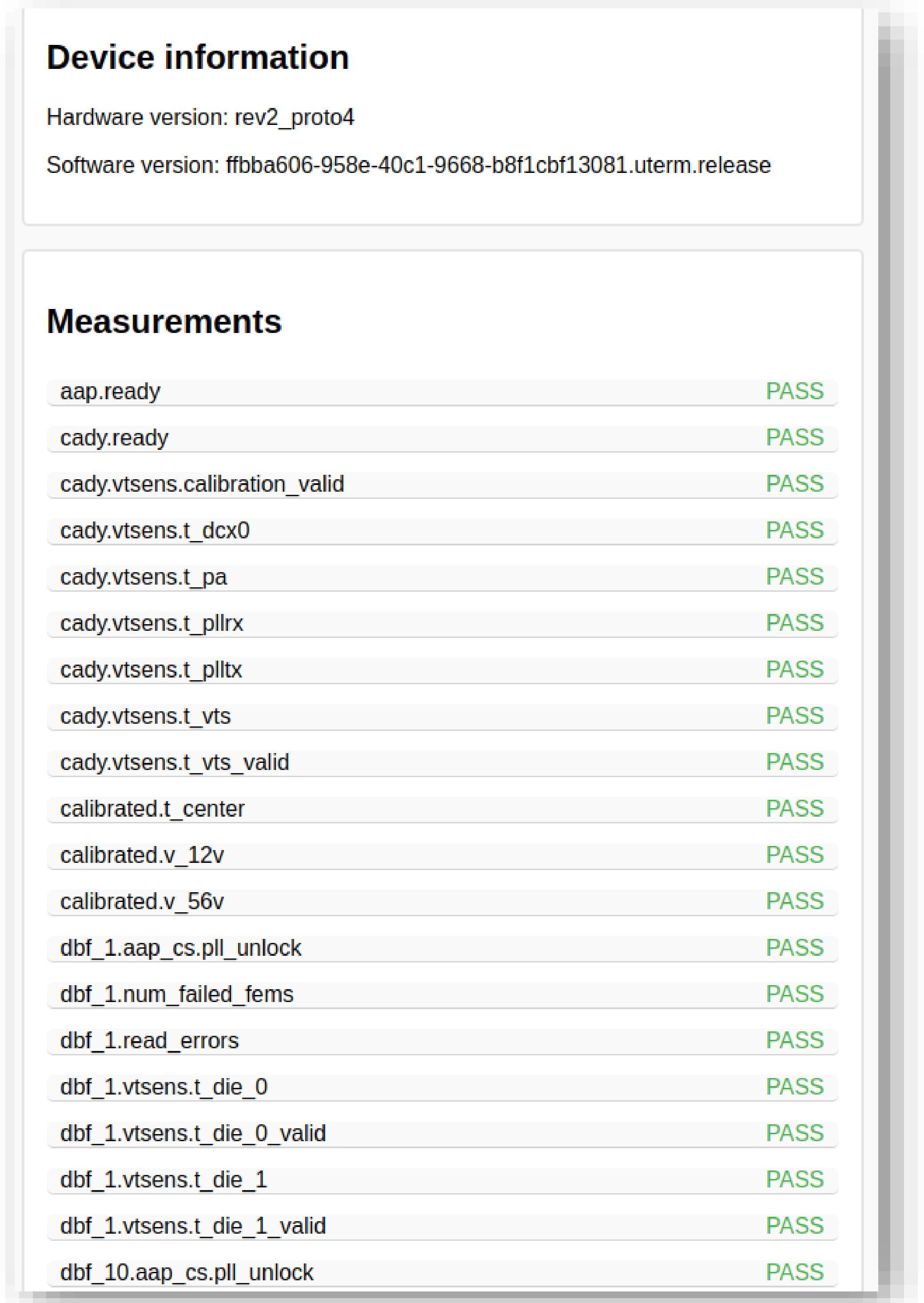
Enable Idle Override

Sky Search

When first enabling sky search, it can take up to 75 seconds for changes to take effect. You can tell when it is active by checking the "Half Duplex State" value in the Us

Note: Sky search and Fast Switching cannot be enabled at the same time

Enable Sky Search Override



Device information

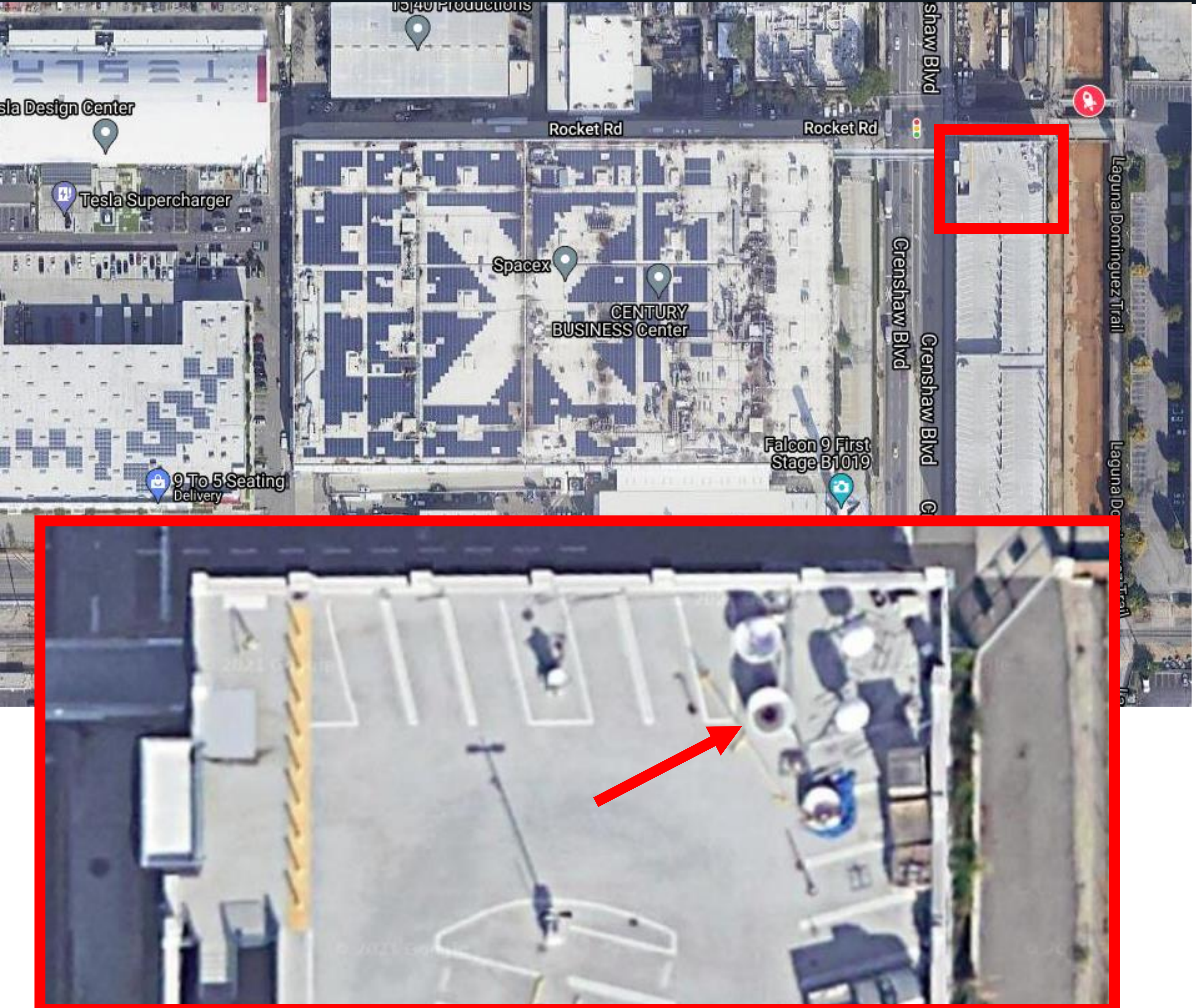
Hardware version: rev2_proto4

Software version: ffbba606-958e-40c1-9668-b8f1cbf13081.uterm.release

Measurements

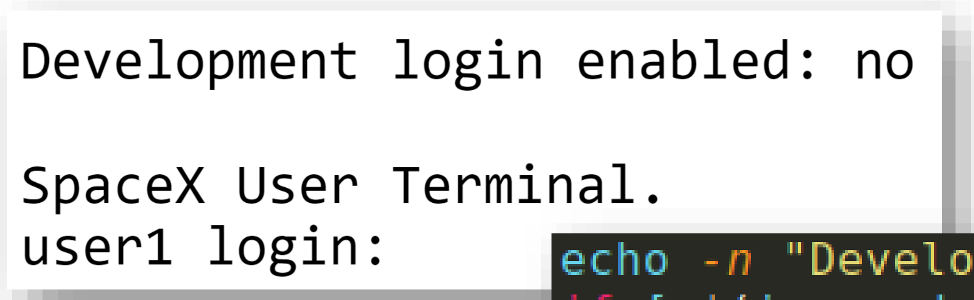
aap.ready	PASS
cady.ready	PASS
cady.vtsens.calibration_valid	PASS
cady.vtsens.t_dcx0	PASS
cady.vtsens.t_pa	PASS
cady.vtsens.t_pllrx	PASS
cady.vtsens.t_plltx	PASS
cady.vtsens.t_vts	PASS
cady.vtsens.t_vts_valid	PASS
calibrated.t_center	PASS
calibrated.v_12v	PASS
calibrated.v_56v	PASS
dbf_1.aap_cs.pll_unlock	PASS
dbf_1.num_failed_fems	PASS
dbf_1.read_errors	PASS
dbf_1.vtsens.t_die_0	PASS
dbf_1.vtsens.t_die_0_valid	PASS
dbf_1.vtsens.t_die_1	PASS
dbf_1.vtsens.t_die_1_valid	PASS
dbf_10.aap_cs.pll_unlock	PASS

Development geofences

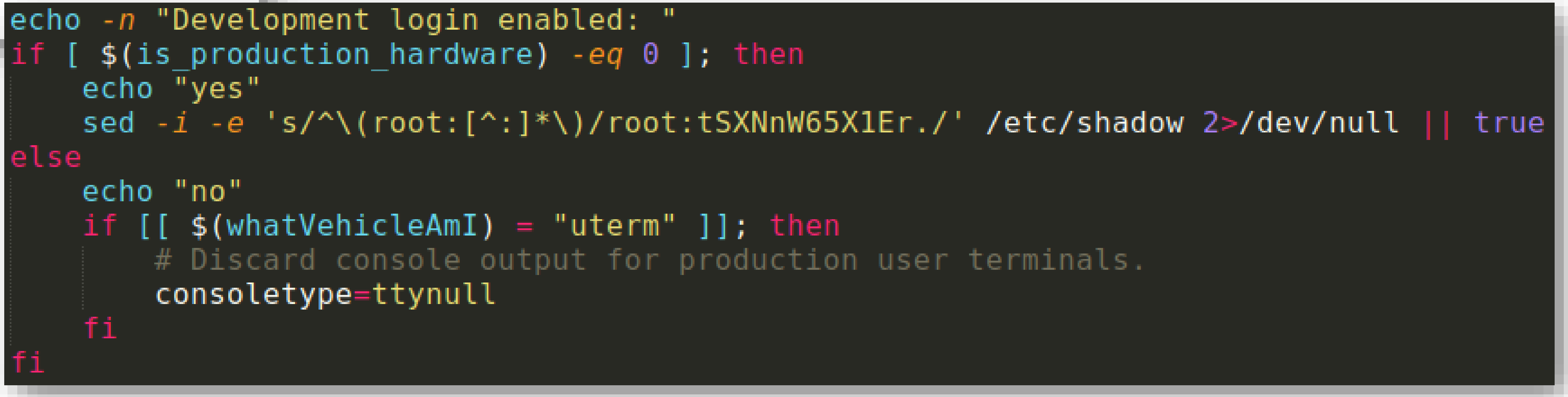


Corson Ave
Corson Ave S
Corson Ave S
Corson Ave S
Connections Museum
Exhibits on the history of the telephone
Monsterhead
Calozzi's Cheeses
Chief Seattle

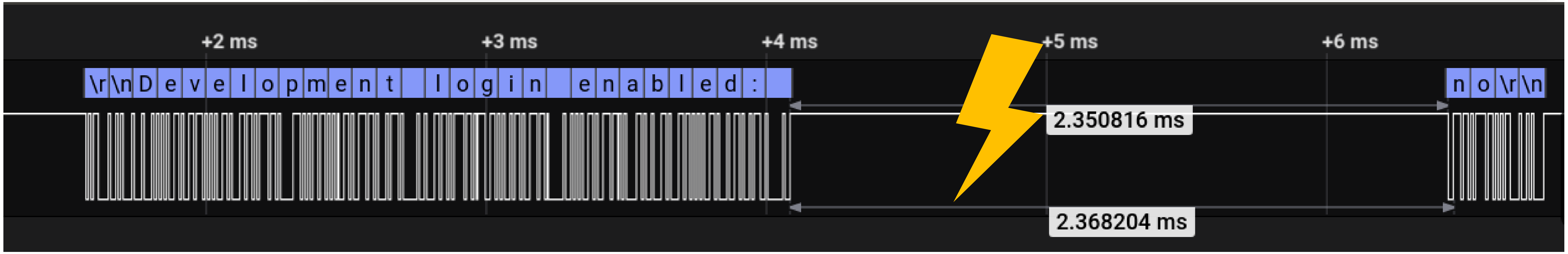
Obtaining root



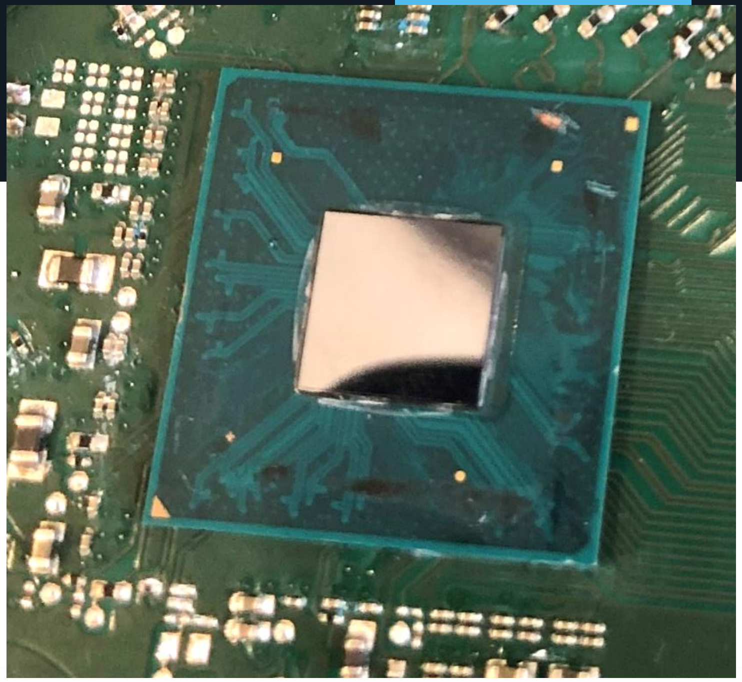
```
Development login enabled: no

SpaceX User Terminal.
user1 login:
```



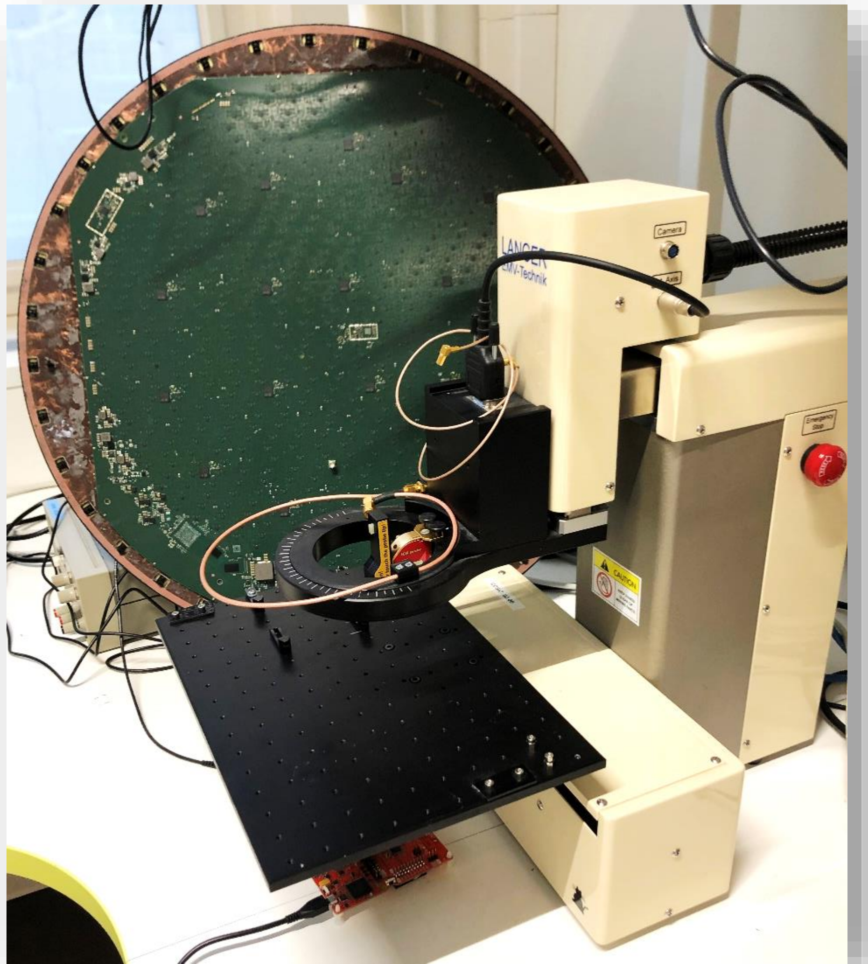
```
echo -n "Development login enabled: "
if [ $(is_production_hardware) -eq 0 ]; then
    echo "yes"
    sed -i -e 's/^\(root:[^:]*\)/root:tSXNnW65X1Er./' /etc/shadow 2>/dev/null || true
else
    echo "no"
    if [[ $(whatVehicleAmI) = "uterm" ]]; then
        # Discard console output for production user terminals.
        consoletype=ttynull
    fi
fi
```

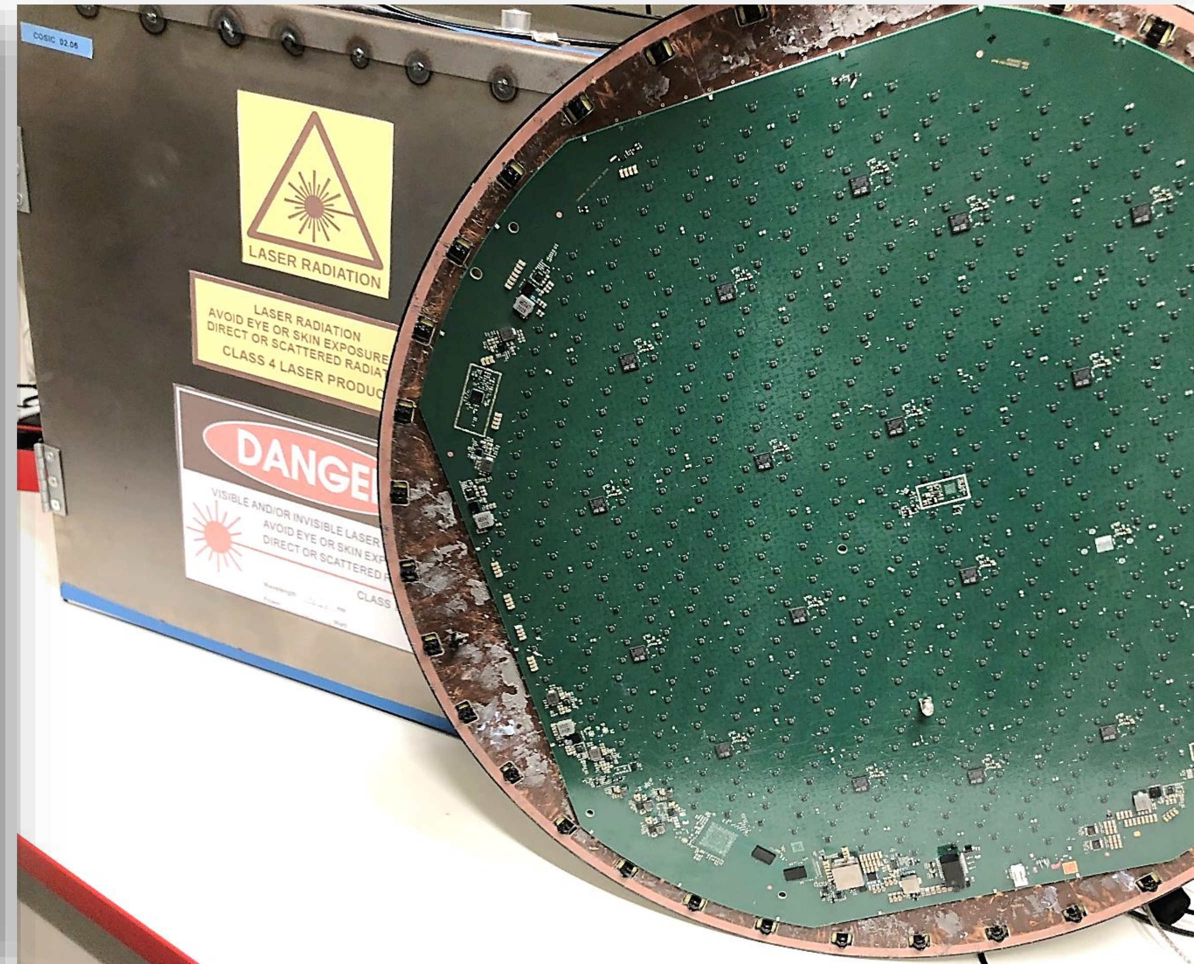


Fault injection



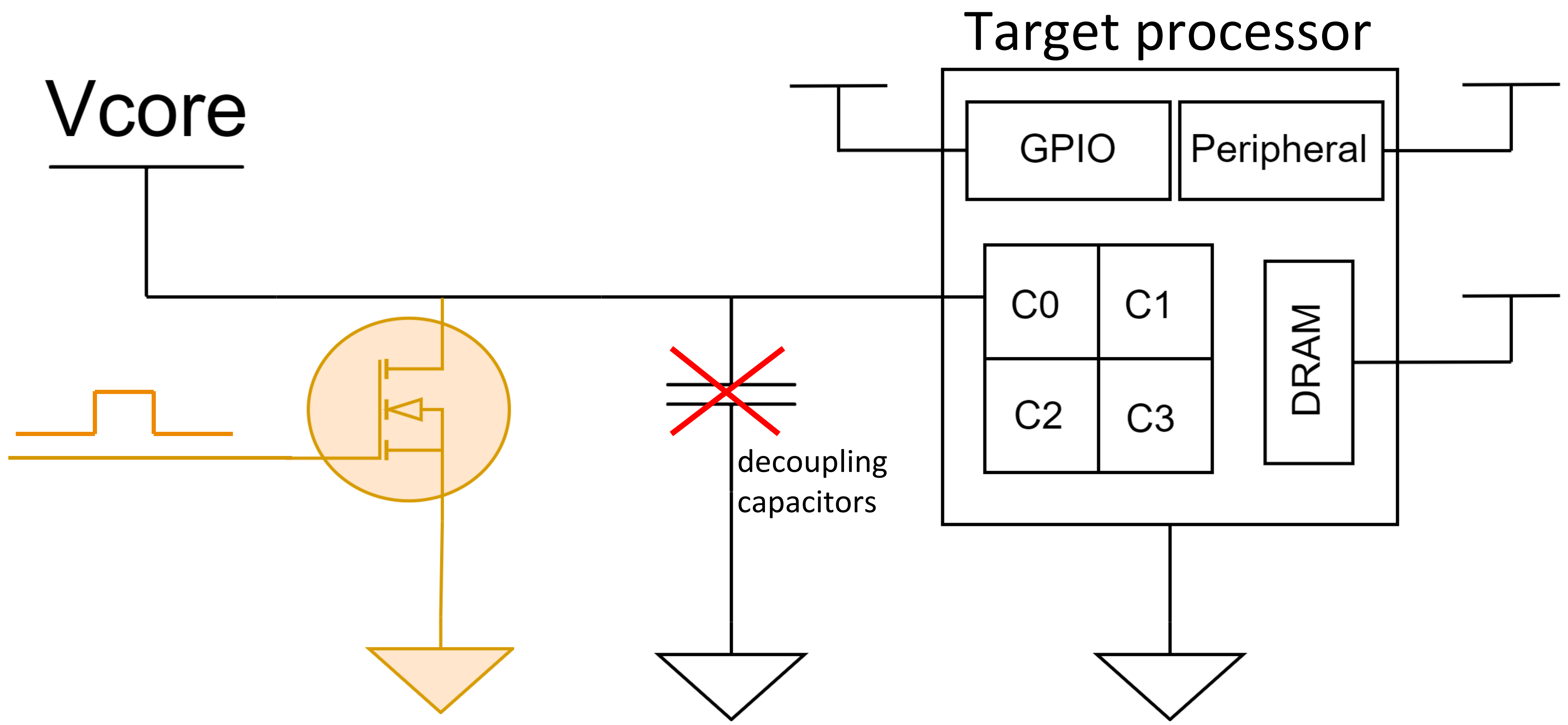
- ✓ Flip-chip packaging exposes die backside
  - Laser Fault Injection, Body Bias Injection, Electromagnetic Fault Injection
- x PCB is too big for our automatic XYZ positioning equipment
  - Likely cumbersome to do on a roof...
- x No development kits





- Differential clock input
  - (But PLL?)
- Reset line
- Voltage Fault Injection

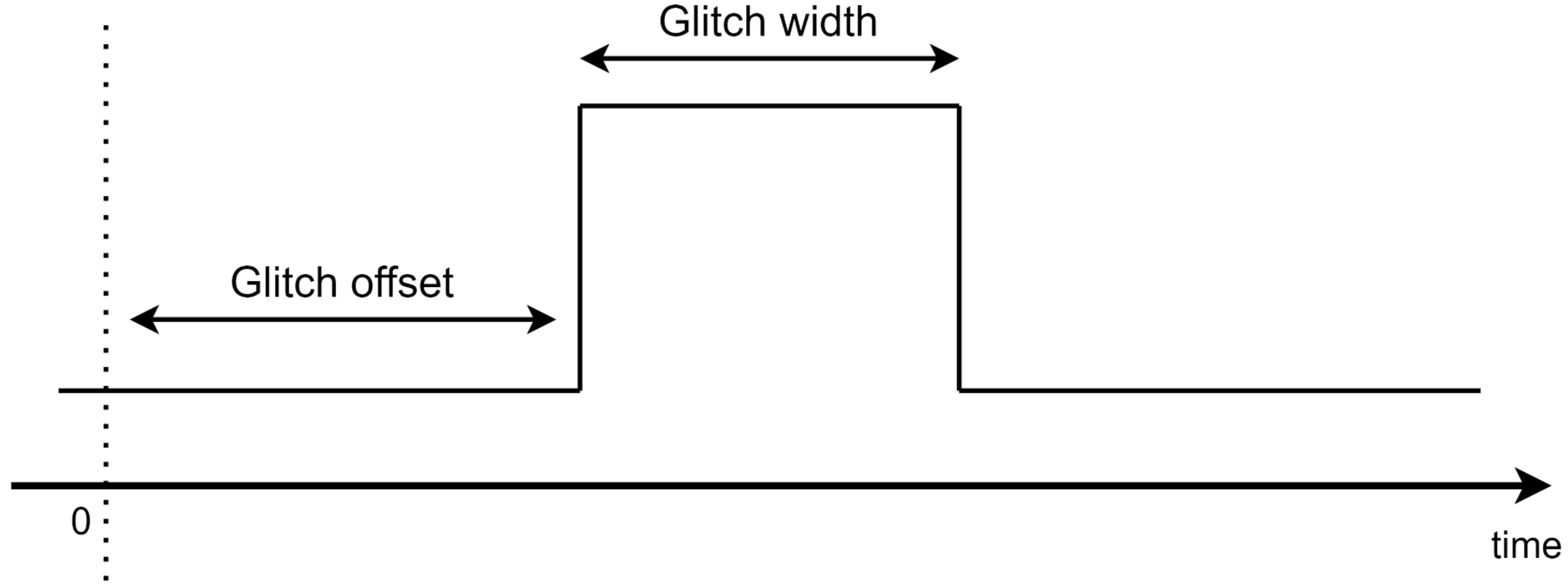
Crowbar VFI


Target processor
Vcore
GPIO
Peripheral
C0
C1
C2
C3
DRAM
decoupling
capacitors

Crowbar VFI: Challenges

- Locate core voltage
  - No schematic and no datasheet → DMM and educated guessing
- Determine which decoupling capacitors to remove
  - None? Some? All?
- Glitch MOSFET selection
  - $R_{DS(on)}$ / drain current / gate threshold V / rise & fall time / turn on & off delay time

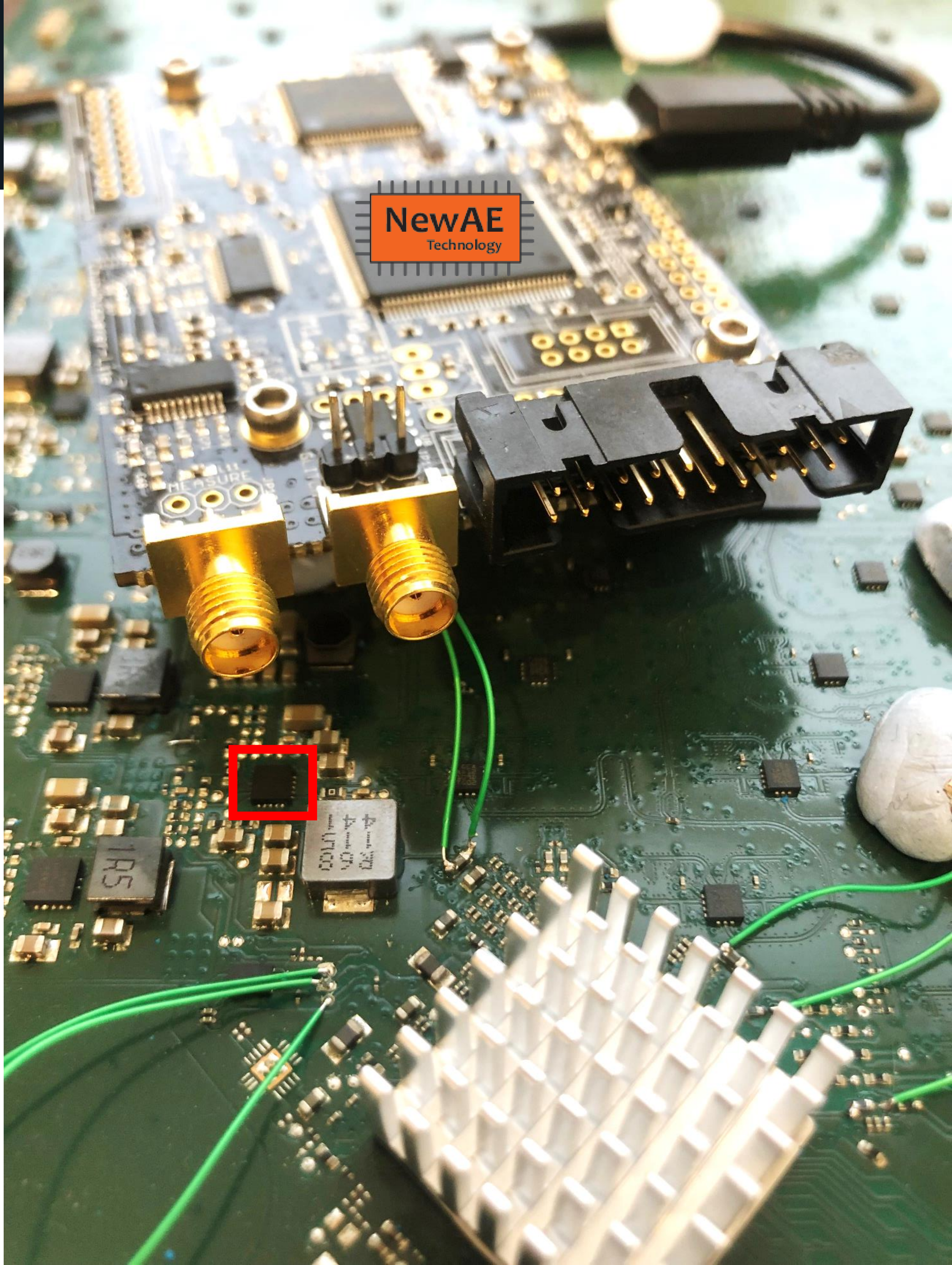
- Finding a suitable trigger signal
  - Time reference
- Determining glitch parameters



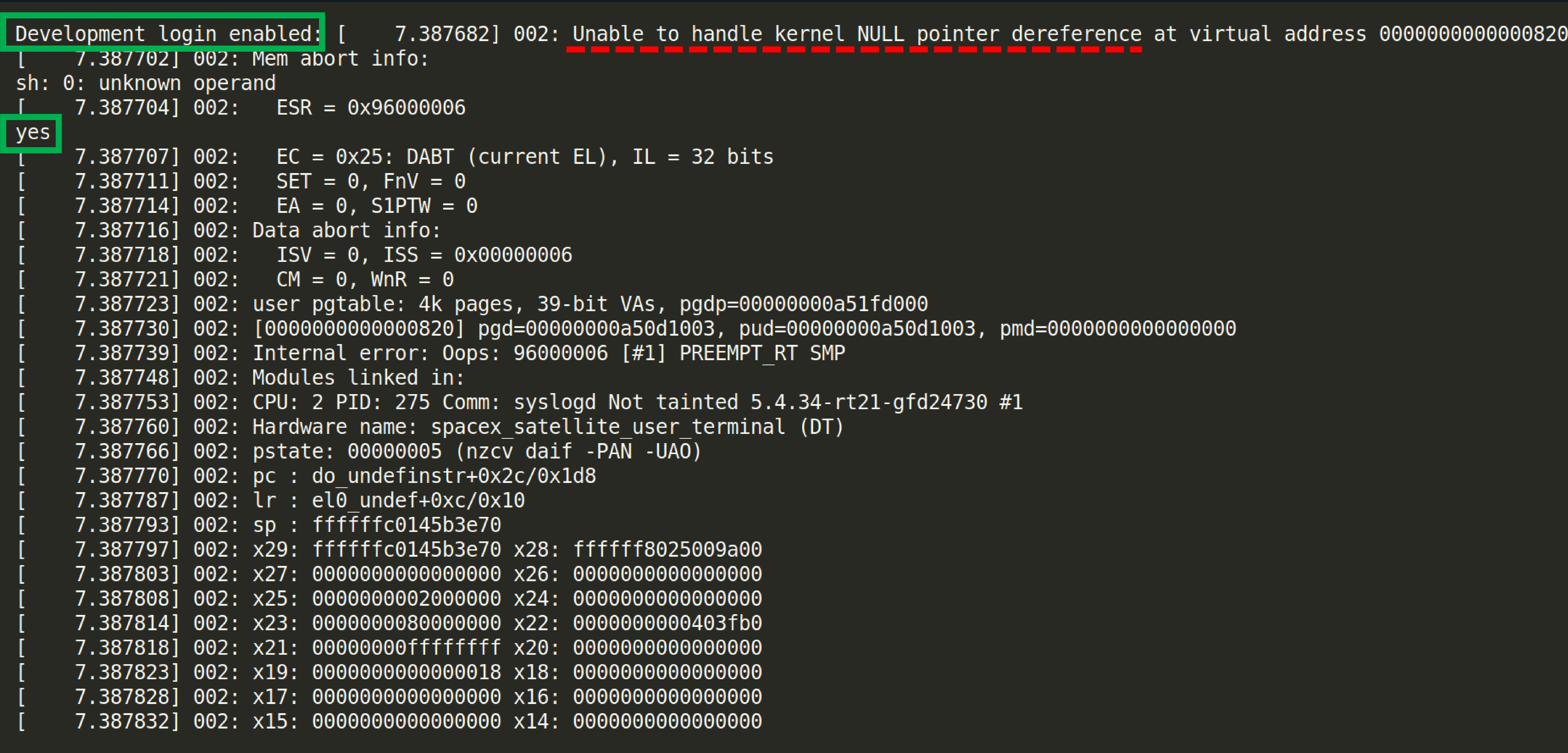
Crowbar VFI

- NewAE ChipWhisperer-Lite (~ $250)
  - Glitch port is connected to the SoC core voltage
  - Momentarily shorts core voltage to GND
- Core voltage:~1V, generated by TI TPS56C230
- All decoupling capacitors untouched at this point!
- Oscilloscope triggers on serial data
  - Trigger output is input to the ChipWhisperer-Lite

- Glitch parameters controlled from Python
  - Offset from trigger point
  - Glitch width



Example output

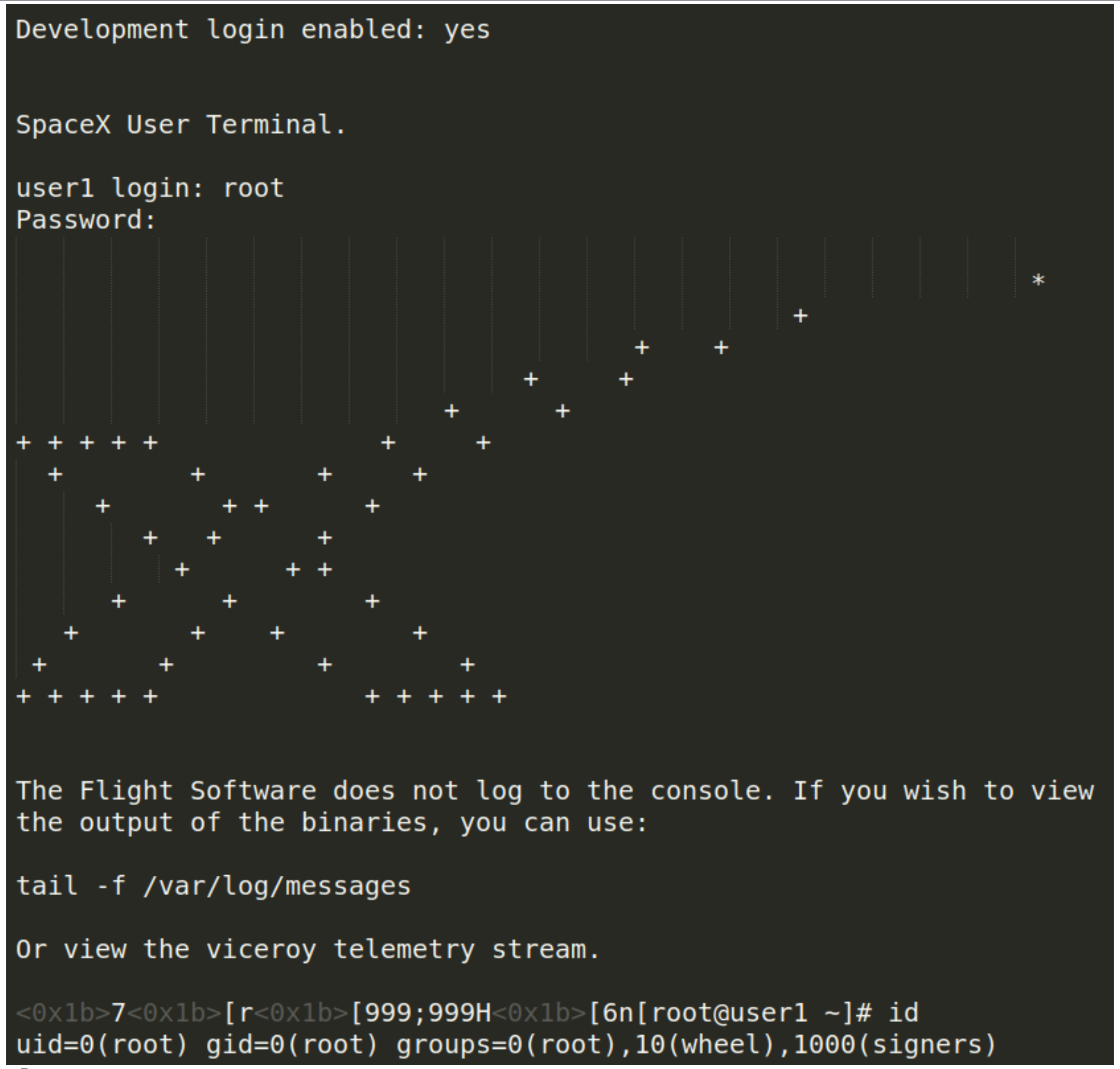


```
Development login enabled: [    7.387682] 002: Unable to handle kernel NULL pointer dereference at virtual address 0000000000000820
[    7.387702] 002: Mem abort info:
sh: 0: unknown operand
[    7.387704] 002:   ESR = 0x96000006
yes
[    7.387707] 002:   EC = 0x25: DABT (current EL), IL = 32 bits
[    7.387711] 002:   SET = 0, FnV = 0
[    7.387714] 002:   EA = 0, S1PTW = 0
[    7.387716] 002: Data abort info:
[    7.387718] 002:   ISV = 0, ISS = 0x00000006
[    7.387721] 002:   CM = 0, WnR = 0
[    7.387723] 002: user pgtable: 4k pages, 39-bit VAs, pgdp=00000000a51fd000
[    7.387730] 002: [0000000000000820] pgd=00000000a50d1003, pud=00000000a50d1003, pmd=0000000000000000
[    7.387739] 002: Internal error: Oops: 96000006 [#1] PREEMPT_RT SMP
[    7.387748] 002: Modules linked in:
[    7.387753] 002: CPU: 2 PID: 275 Comm: syslogd Not tainted 5.4.34-rt21-gfd24730 #1
[    7.387760] 002: Hardware name: spacex_satellite_user_terminal (DT)
[    7.387766] 002: pstate: 00000005 (nzcv daif -PAN -UAO)
[    7.387770] 002: pc : do_undefinstr+0x2c/0x1d8
[    7.387787] 002: lr : el0_undef+0xc/0x10
[    7.387793] 002: sp : ffffffc0145b3e70
[    7.387797] 002: x29: ffffffc0145b3e70 x28: ffffff8025009a00
[    7.387803] 002: x27: 0000000000000000 x26: 0000000000000000
[    7.387808] 002: x25: 0000000002000000 x24: 0000000000000000
[    7.387814] 002: x23: 0000000080000000 x22: 0000000000403fb0
[    7.387818] 002: x21: 00000000ffffffff x20: 0000000000000000
[    7.387823] 002: x19: 0000000000000018 x18: 0000000000000000
[    7.387828] 002: x17: 0000000000000000 x16: 0000000000000000
[    7.387832] 002: x15: 0000000000000000 x14: 0000000000000000
```

Results

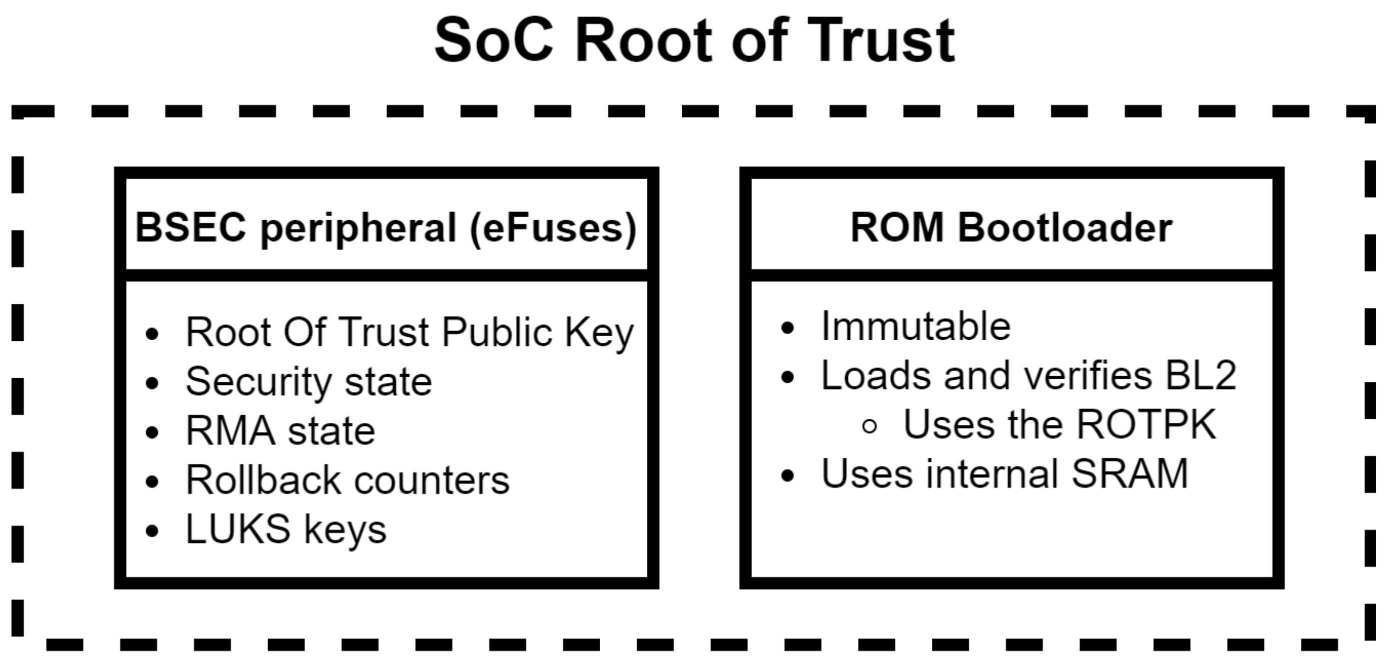
- ✓ The Proof-of-Concept works
  - ✓ Was reproduced by the SpaceX PSIRT
- ✓ Easy to produce (undesirable) faults
  - ✓ A fully booted SoC is already being pushed to its limits

- x Slow: 1 attempt every 12 seconds (one per boot)
  - x Low success rate: many hours for one good attempt
- x Unreliable: successful glitch often also results in other errors



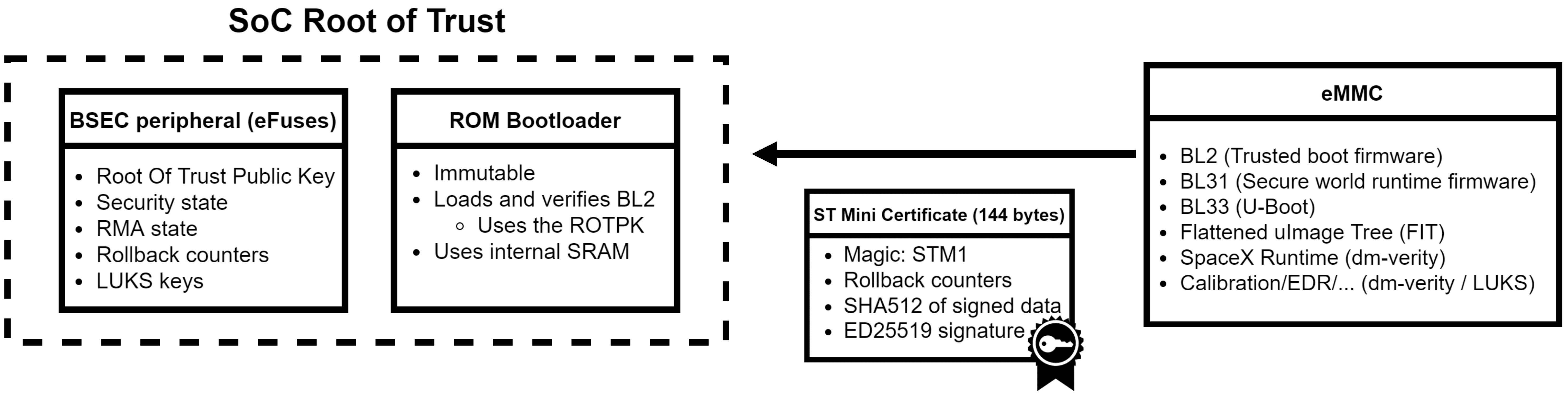
```
Development login enabled: yes


SpaceX User Terminal.

user1 login: root
Password:
                                                                *
                                                 +
                                       +    +
                                +     +
                           +      +
+ + + + +              +     +
  +        +       +     +
     +       + +      +
        +   +      +
          +      + +
      +      +        +
   +       +    +        +
 +       +         +        +
+ + + + +             + + + + +


The Flight Software does not log to the console. If you wish to view
the output of the binaries, you can use:

tail -f /var/log/messages

Or view the viceroy telemetry stream.

<0x1b>7<0x1b>[r<0x1b>[999;999H<0x1b>[6n[root@user1 ~]# id
uid=0(root) gid=0(root) groups=0(root),10(wheel),1000(signers)
```

STM/SpaceX ARM TFA-A



- In theory everything outside of the SoC should be considered untrusted
- ROTPK and Security state fuses are blown during manufacturing
  - It *should* be impossible to revert a blown fuse
  - Care must be taken when blowing additional fuses
- Security state fuses enable/disable debugging access

STM/SpaceX ARM TFA-A



1. BL1 loads BL2 certificate from eMMC
2. BL1 verifies the certificate's signature
3. BL1 loads the BL2 firmware from eMMC
4. BL1 verifies that SHA512(BL2) matches the hash contained in the certificate

Challenges

- We do not have access to documentation or open/unfused samples
  - Commonly: develop the attack on a development board first
- We can’t run our own test program
  - Commonly: nested loop counter with observable output and GPIO pin for triggering
- We can’t dump and reverse engineer the ROM bootloader
  - (We will dump it later ;))
  - We don’t really know what operation is happening when
  - But we do know the later stages are based on TF-A

Tricks of the trade

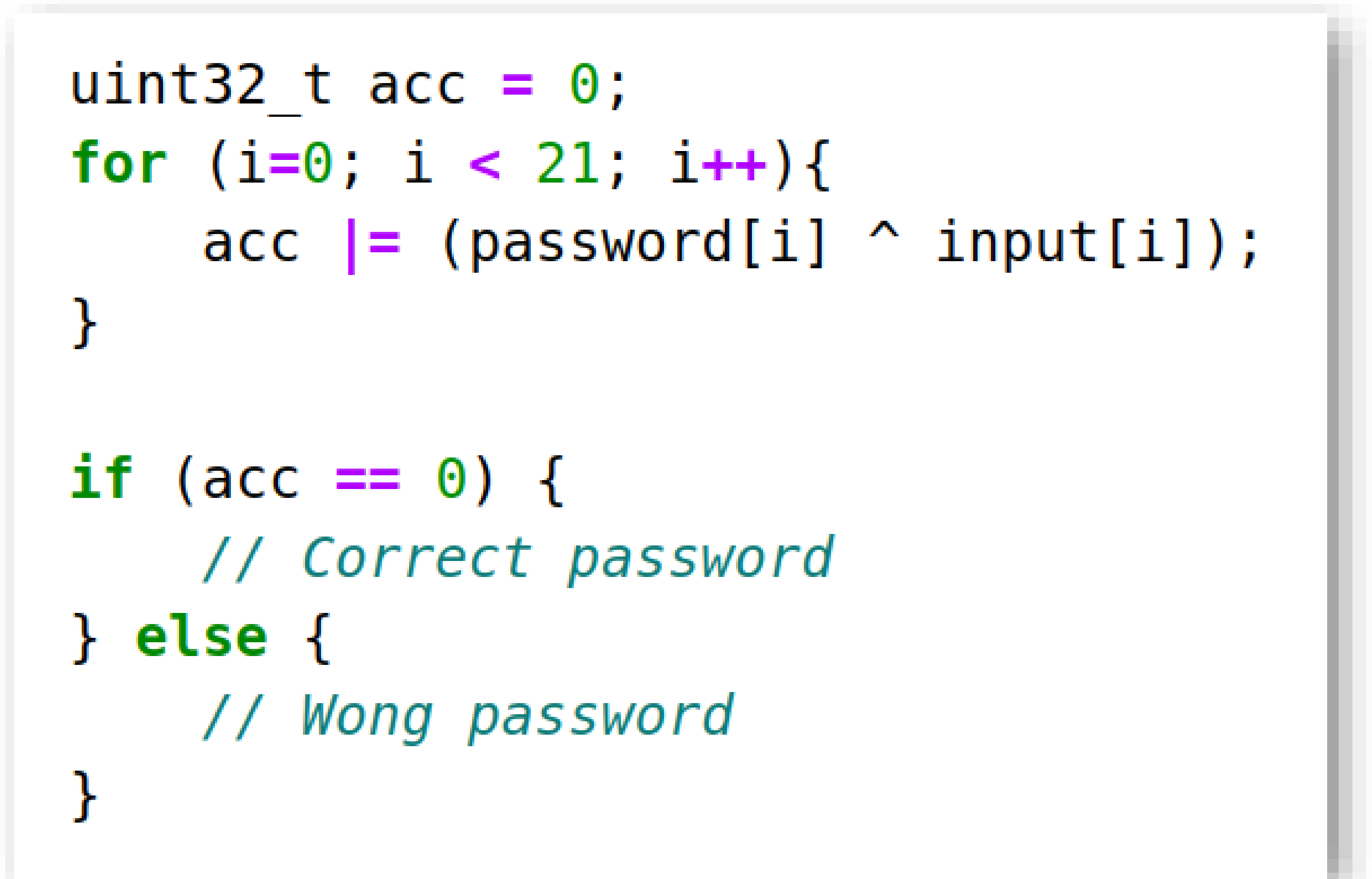
- Boot with a second stage that is invalid and observe differences
  - Invalid certificate signature
  - Invalid bootloader hash in the certificate
  - Valid certificate with a bootloader that does not match the certificate

- Attempt to glitch a valid certificate into a signature verification failure!
  - Allows to determine a suitable range of glitch widths

- Side-channels!
  - Power consumption, EM emanations, timing differences, temperature, …

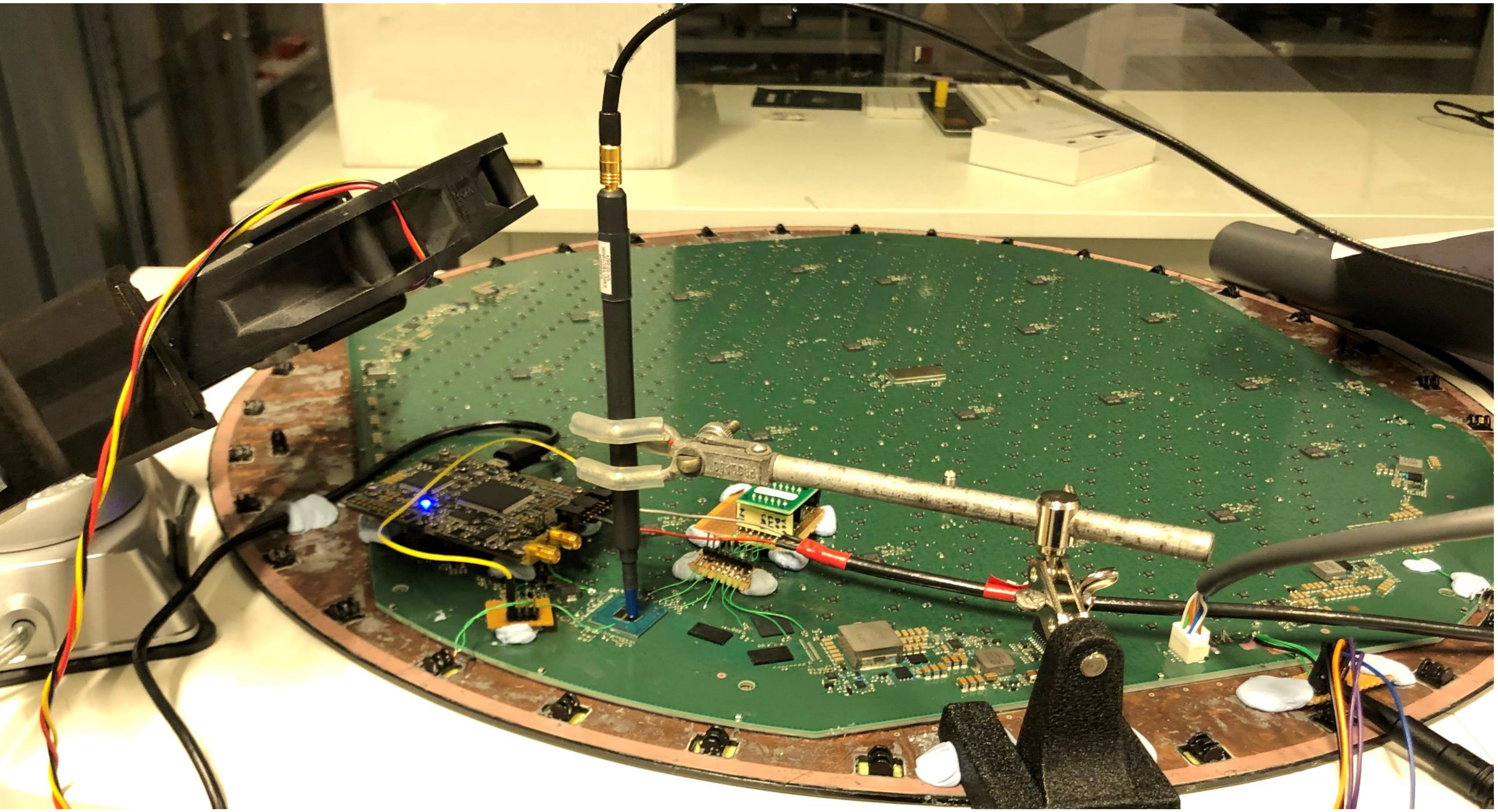
Tricks of the trade

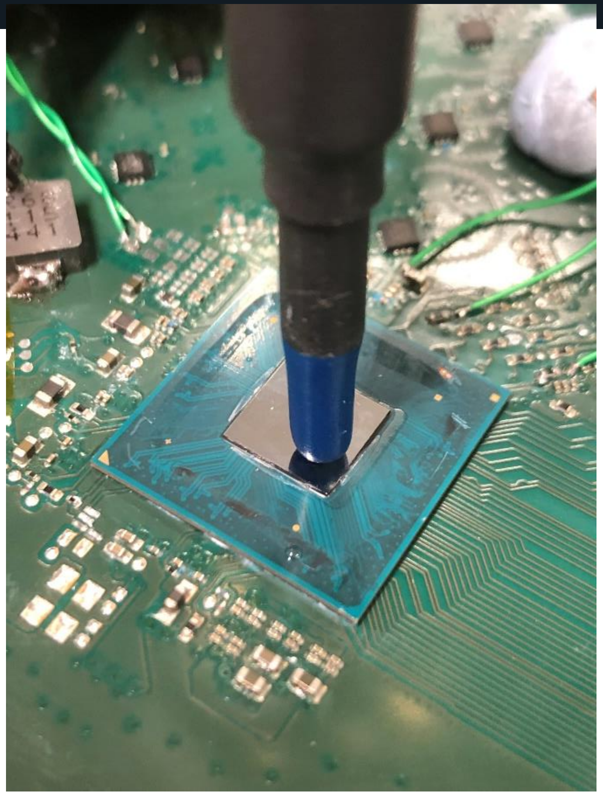
- Hardware and the software executing on it can be susceptible to glitching in multiple unexpected ways!
  - Your (and the developer’s) mental model is likely incorrect
  - Try to be exhaustive and do not simply try to glitch at the end of an operation

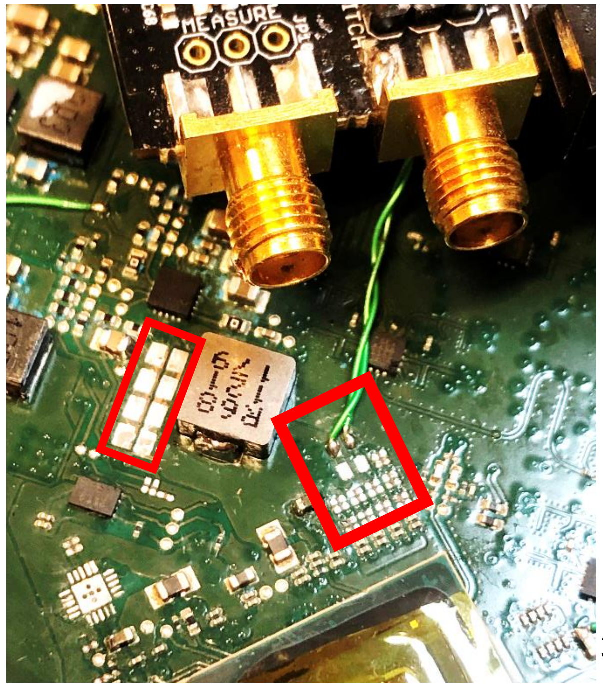


```
uint32_t acc = 0;
for (i=0; i < 21; i++){
    acc |= (password[i] ^ input[i]);
}

if (acc == 0) {
    // Correct password
} else {
    // Wong password
}
```

BL1 Glitch setup

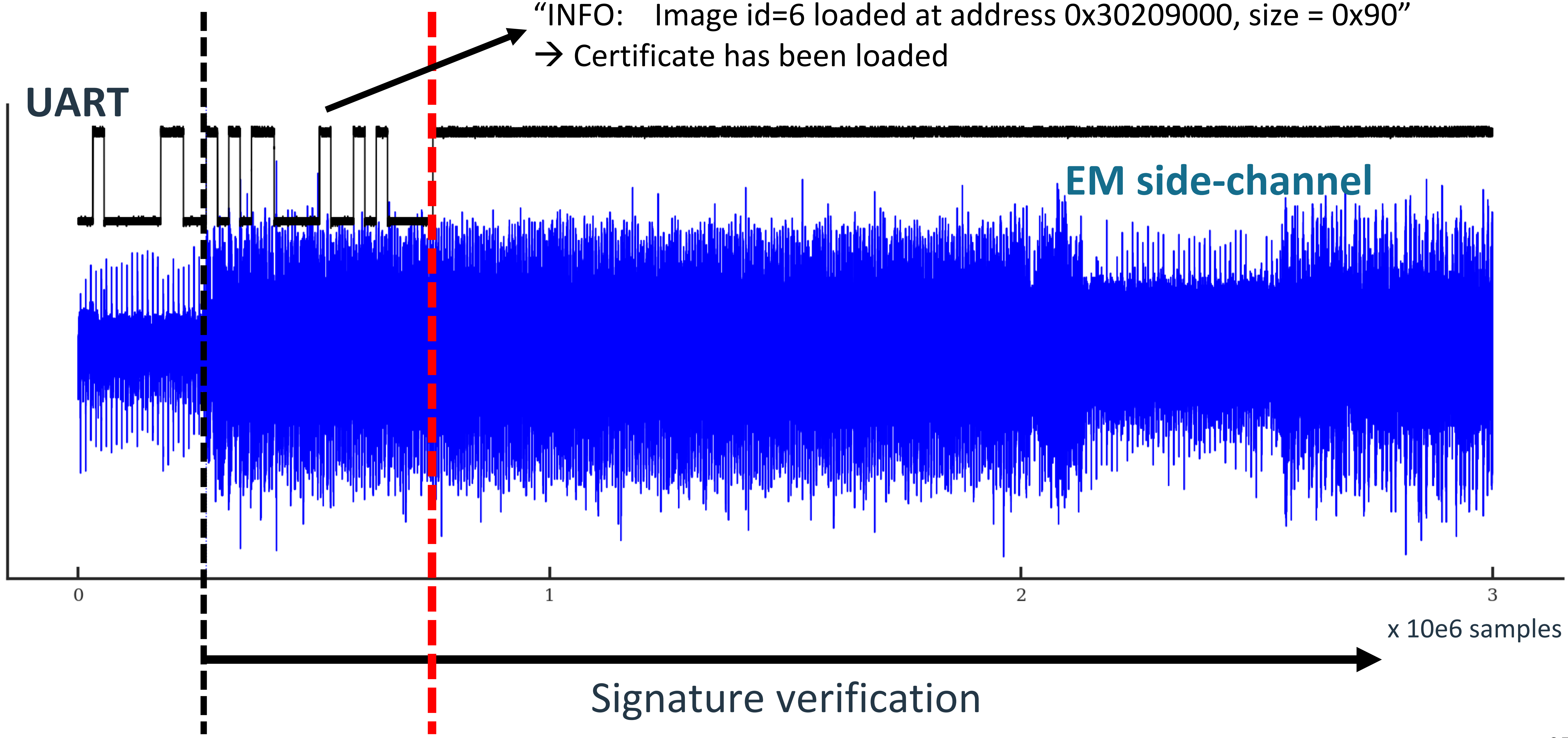




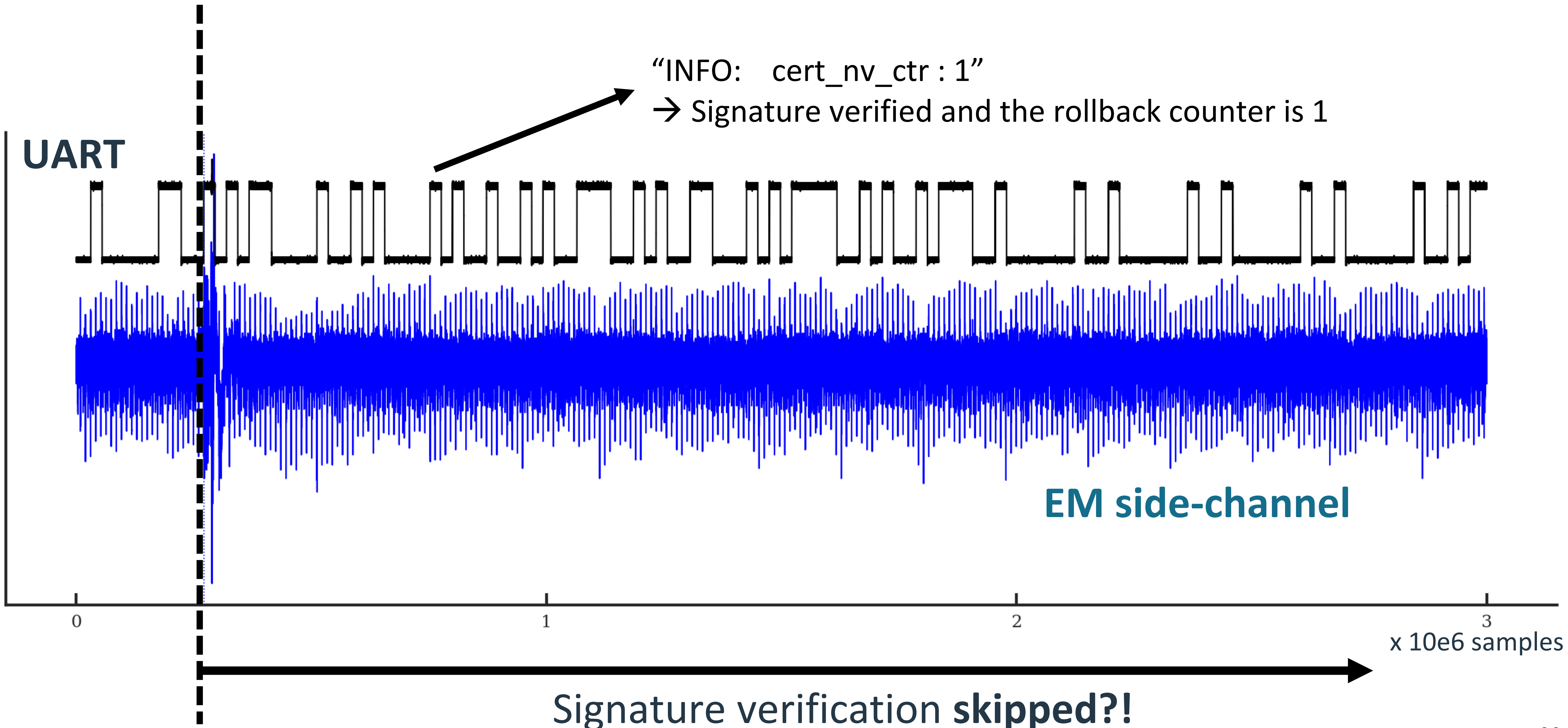


Normal boot

“INFO: Image id=6 loaded at address 0x30209000, size = 0x90”
→ Certificate has been loaded



Glitched boot



Signature verification **skipped?!**

ROM Bootloader (BL1)

- Mapped at 0x30000000 and readable from BL2!
  - BSEC eFuses mapped at 0x22400000 (shadow registers)

- Emulated the ROM bootloader using Unicorn Engine
  - Fuzzed using AFL++ in Unicorn mode
    - No bugs found so far…

- Simulated instruction skip faults in Unicorn Engine
    - Single instruction skip faults do not result in the observed behavior!
      - Code has some control flow checks and redundant operations
    - Skipping two consecutive instructions does result in the observed behavior
      - (Actual fault model is likely to be different)

github.com/unicorn-engine/unicorn

github.com/AFLplusplus/AFLplusplus

BL1 glitch detection example

BL1 UART output

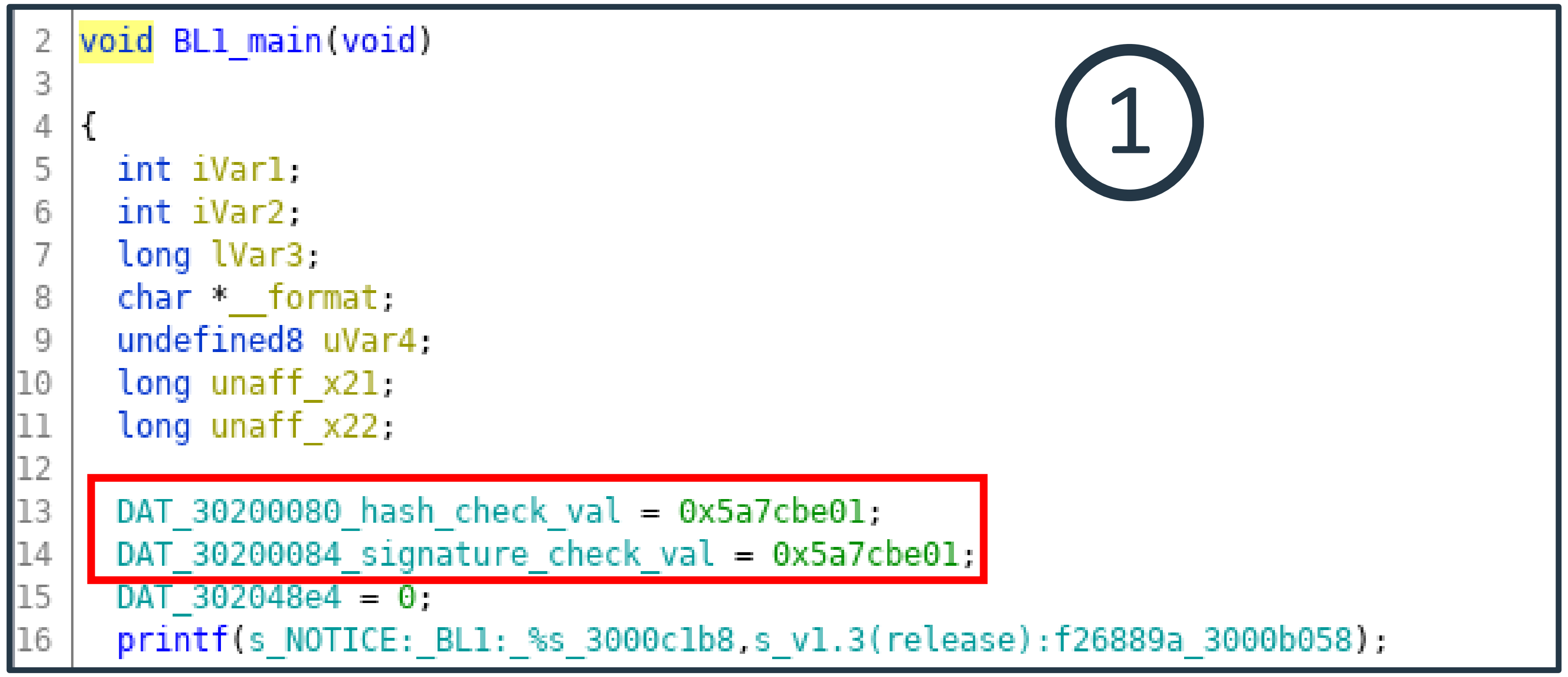
```
INFO:    BL1: Get the image descriptor
INFO:    BL1: Loading BL2
INFO:    Loading image id=6 at address 0x30209000
INFO:    Skip reserving region [base = 0x30209000, size = 0x90]
INFO:    Image id=6 loaded at address 0x30209000, size = 0x90

INFO:    cert_nv_ctr : 1
INFO:    plat_nv_ctr : 0
INFO:    Loading image id=1 at address 0x30209000
INFO:    Image id=1 loaded at address 0x30209000, size = 0xf178

NOTICE:  BL1: Booting BL2
NOTICE:  plat_error_handler      err = -80
INFO:    Authentication error !!!
```

Certificate has been loaded
Contains invalid signature but
valid digest of BL2 firmware

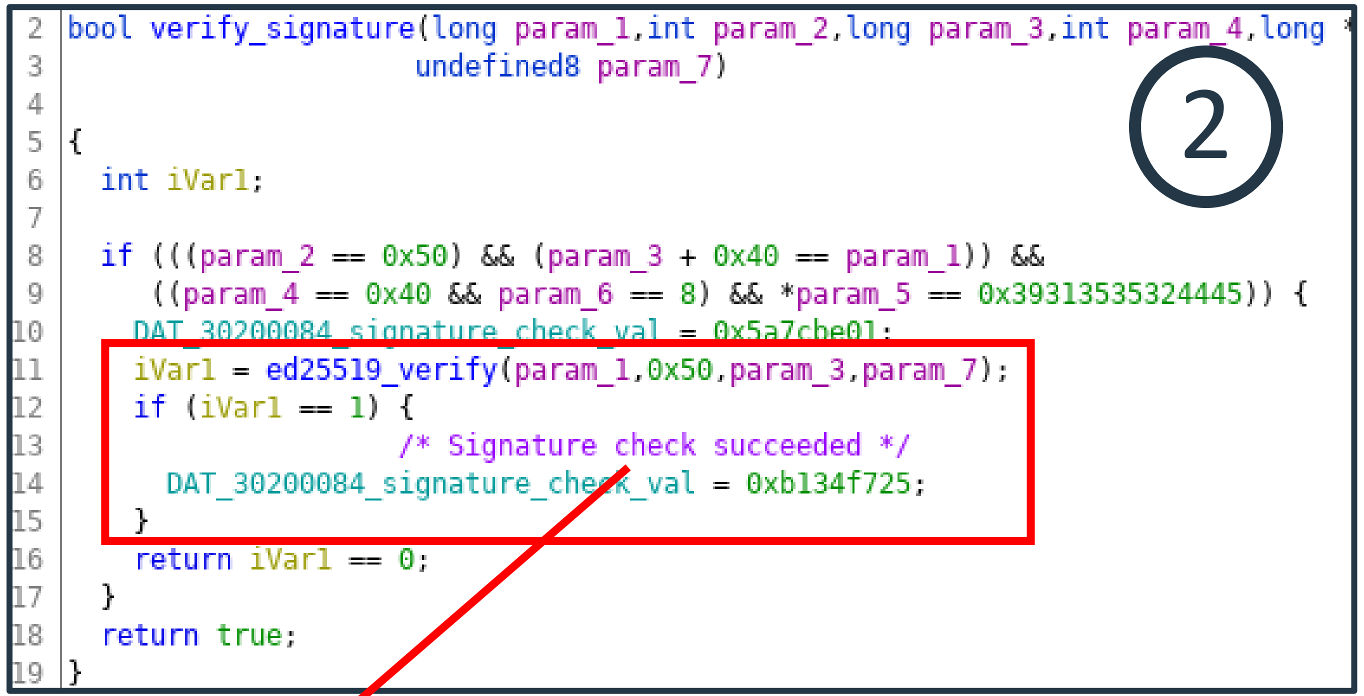
Signature verification succeeded!
Loaded BL2 firmware and
verified hash digest

Final control flow check detects
our glitch! ☹

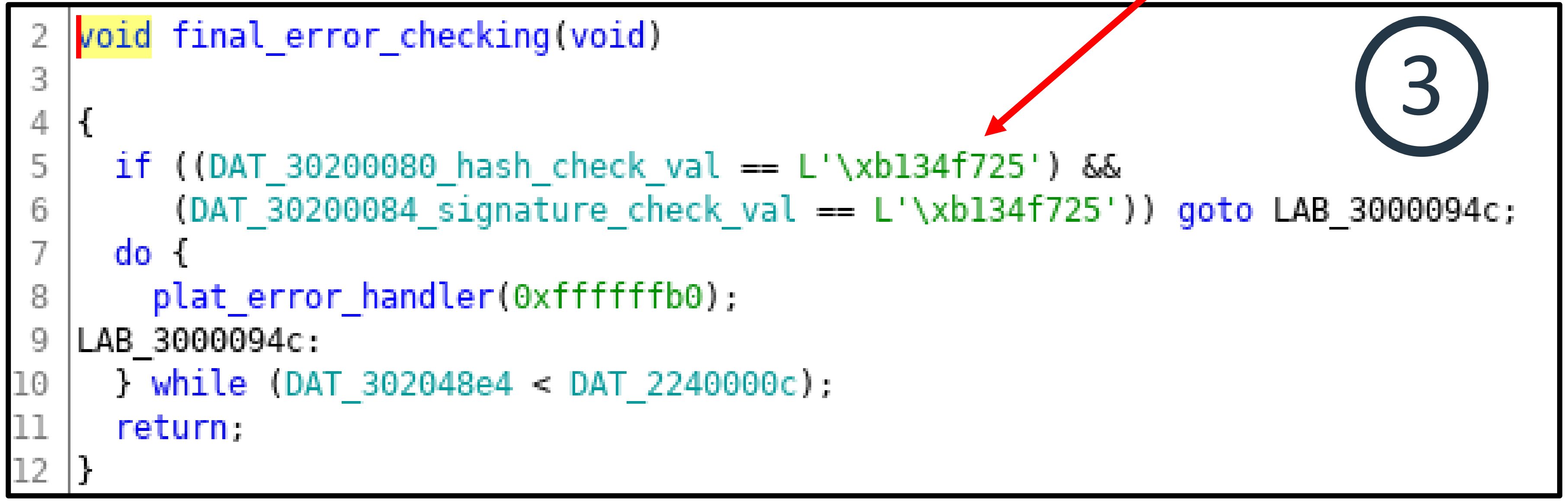
BL1 glitch detection example



```
void BL1_main(void)

{
  int iVar1;
  int iVar2;
  long lVar3;
  char *__format;
  undefined8 uVar4;
  long unaff_x21;
  long unaff_x22;

  DAT_30200080_hash_check_val = 0x5a7cbe01;
  DAT_30200084_signature_check_val = 0x5a7cbe01;
  DAT_302048e4 = 0;
  printf(s_NOTICE:_BL1:_%s_3000c1b8,s_v1.3(release):f26889a_3000b058);
```



```
bool verify_signature(long param_1,int param_2,long param_3,int param_4,long *
                      undefined8 param_7)

{
  int iVar1;

  if ((((param_2 == 0x50) && (param_3 + 0x40 == param_1)) &&
     ((param_4 == 0x40 && param_6 == 8) && *param_5 == 0x39313535324445)) {
    DAT_30200084_signature_check_val = 0x5a7cbe01;
    iVar1 = ed25519_verify(param_1,0x50,param_3,param_7);
    if (iVar1 == 1) {
                    /* Signature check succeeded */
      DAT_30200084_signature_check_val = 0xb134f725;
    }
    return iVar1 == 0;
  }
  return true;
}
```

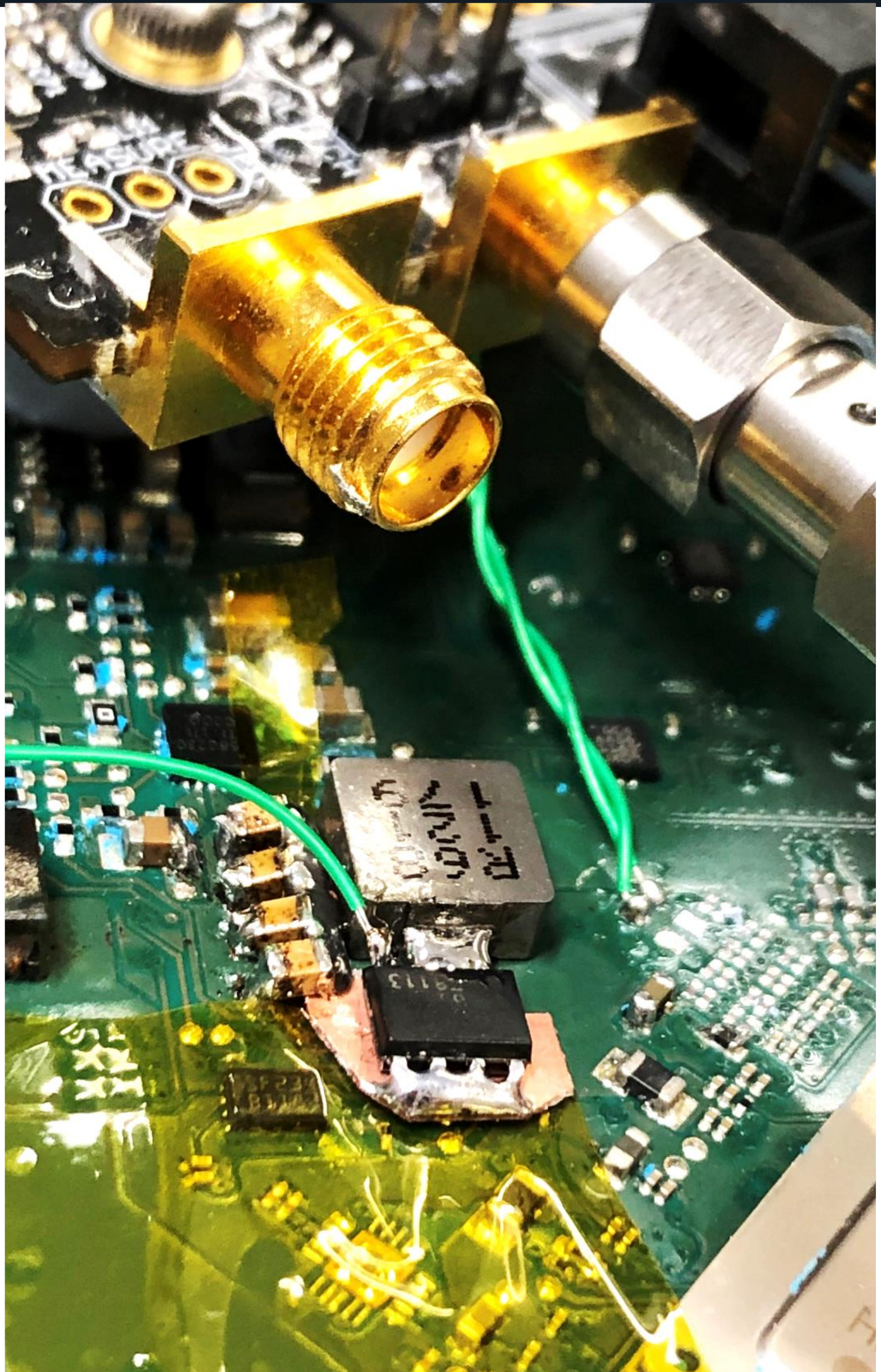


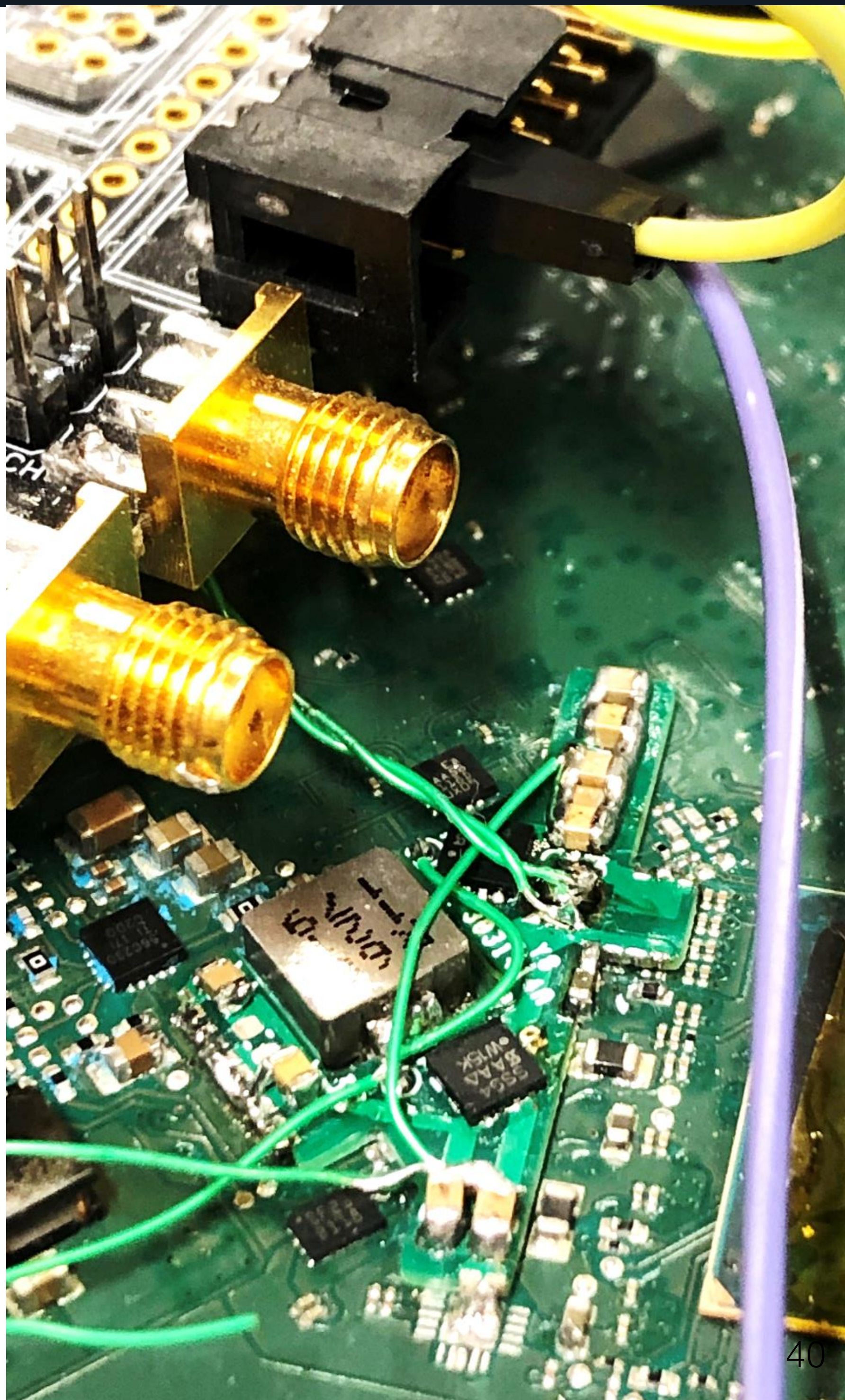
```
void final_error_checking(void)

{
  if ((DAT_30200080_hash_check_val == L'\xb134f725') &&
     (DAT_30200084_signature_check_val == L'\xb134f725')) goto LAB_3000094c;
  do {
    plat_error_handler(0xffffffb0);
LAB_3000094c:
  } while (DAT_302048e4 < DAT_2240000c);
  return;
}
```

Called right before passing control to BL2

Enabling decoupling capacitors

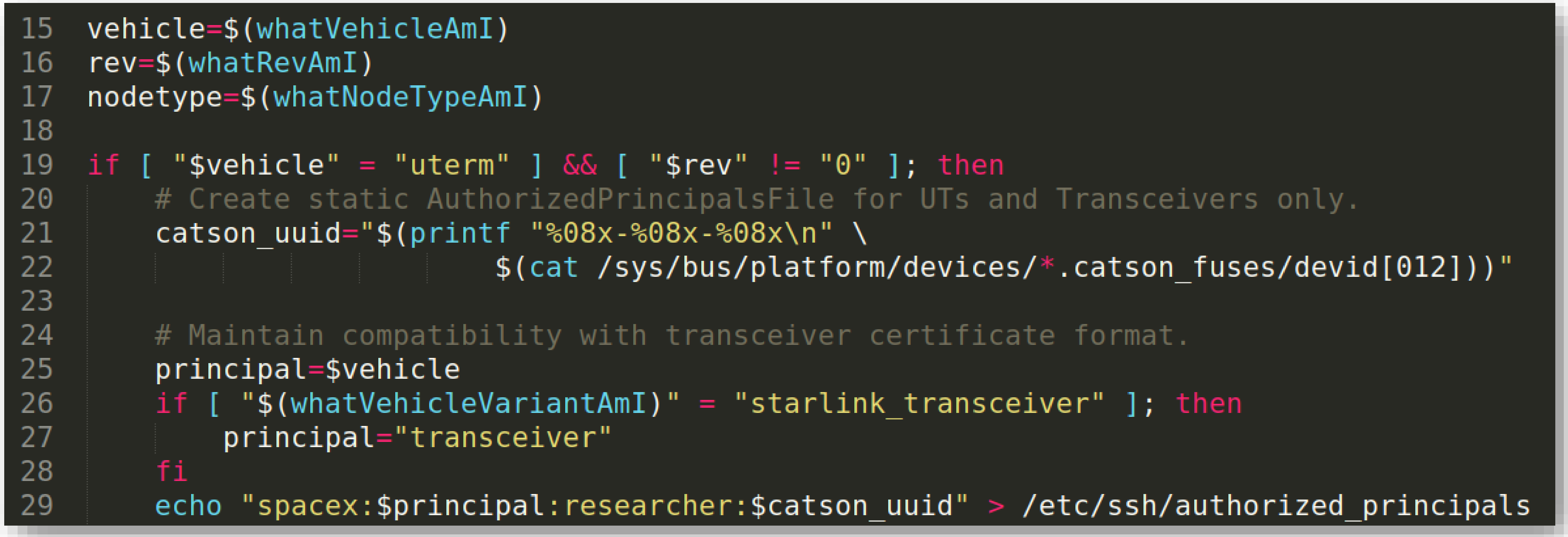
- Decoupling capacitors are needed for later boot stages
- Experimented with:
  - N-channel MOSFETS
  - P-channel MOSFETS
  - High/Low side switching
  - Gate voltage
  - MOSFET drivers
  - Capacitor sizes
  - Timing





Researcher access

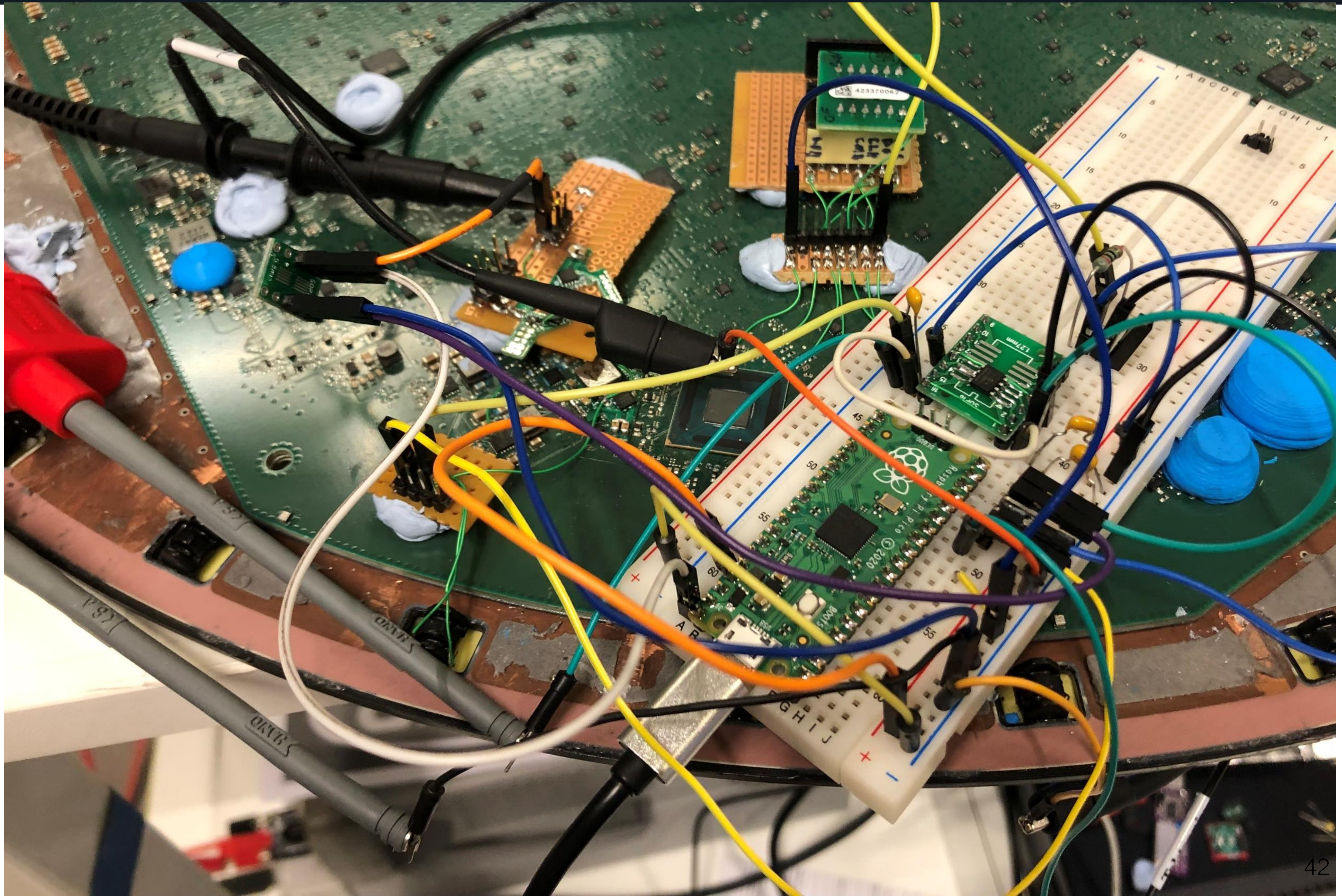
- Demonstrated a full attack in the lab!
  - But the setup is still too bulky to be used in a practical setting (e.g., on a roof)
- SpaceX offered an easy way out: SSH access through a Yubikey
  - But I was already too far down the rabbit hole …



```
vehicle=$(whatVehicleAmI)
rev=$(whatRevAmI)
nodetype=$(whatNodeTypeAmI)

if [ "$vehicle" = "uterm" ] && [ "$rev" != "0" ]; then
    # Create static AuthorizedPrincipalsFile for UTs and Transceivers only.
    catson_uuid="$(printf "%08x-%08x-%08x\n" \
                        $(cat /sys/bus/platform/devices/*.catson_fuses/devid[012]))"

    # Maintain compatibility with transceiver certificate format.
    principal=$vehicle
    if [ "$(whatVehicleVariantAmI)" = "starlink_transceiver" ]; then
        principal="transceiver"
    fi
    echo "spacex:$principal:researcher:$catson_uuid" > /etc/ssh/authorized_principals
```

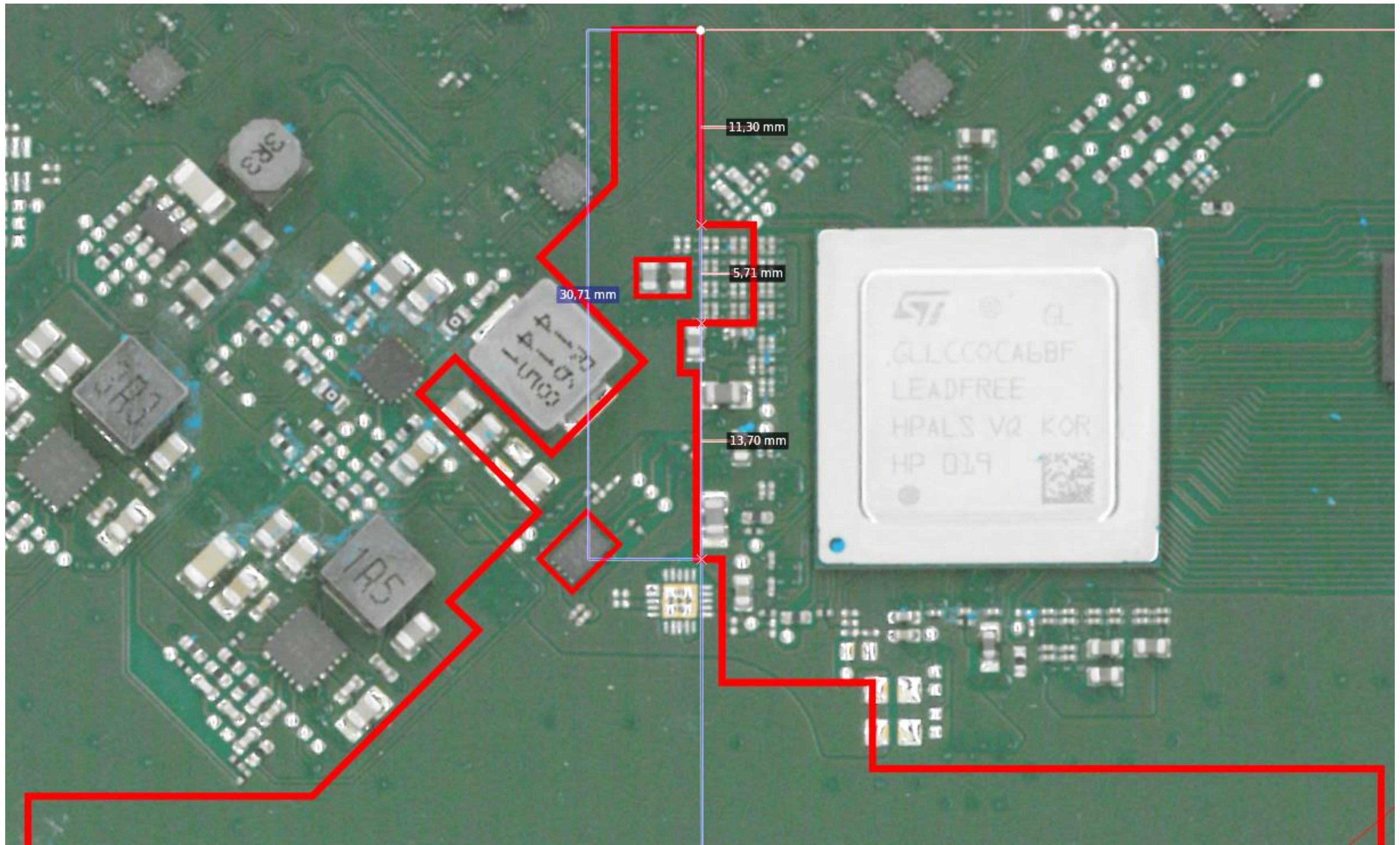
Creating a mobile setup

- Replacing lab equipment with low-cost off-the-shelf components
- RPI Pico replaces oscilloscope and ChipWhisperer
- Works
  - But still messy…



PCB design

- Scanner @ 600 DPI
- Draw board outline at real size in Inkscape
    - Load in KiCad and use in the edgecuts layer



Modchip

Castellated holes to mount to the UT PCB

Glitch/crowbar MOSFET

Decoupling MOSFETs

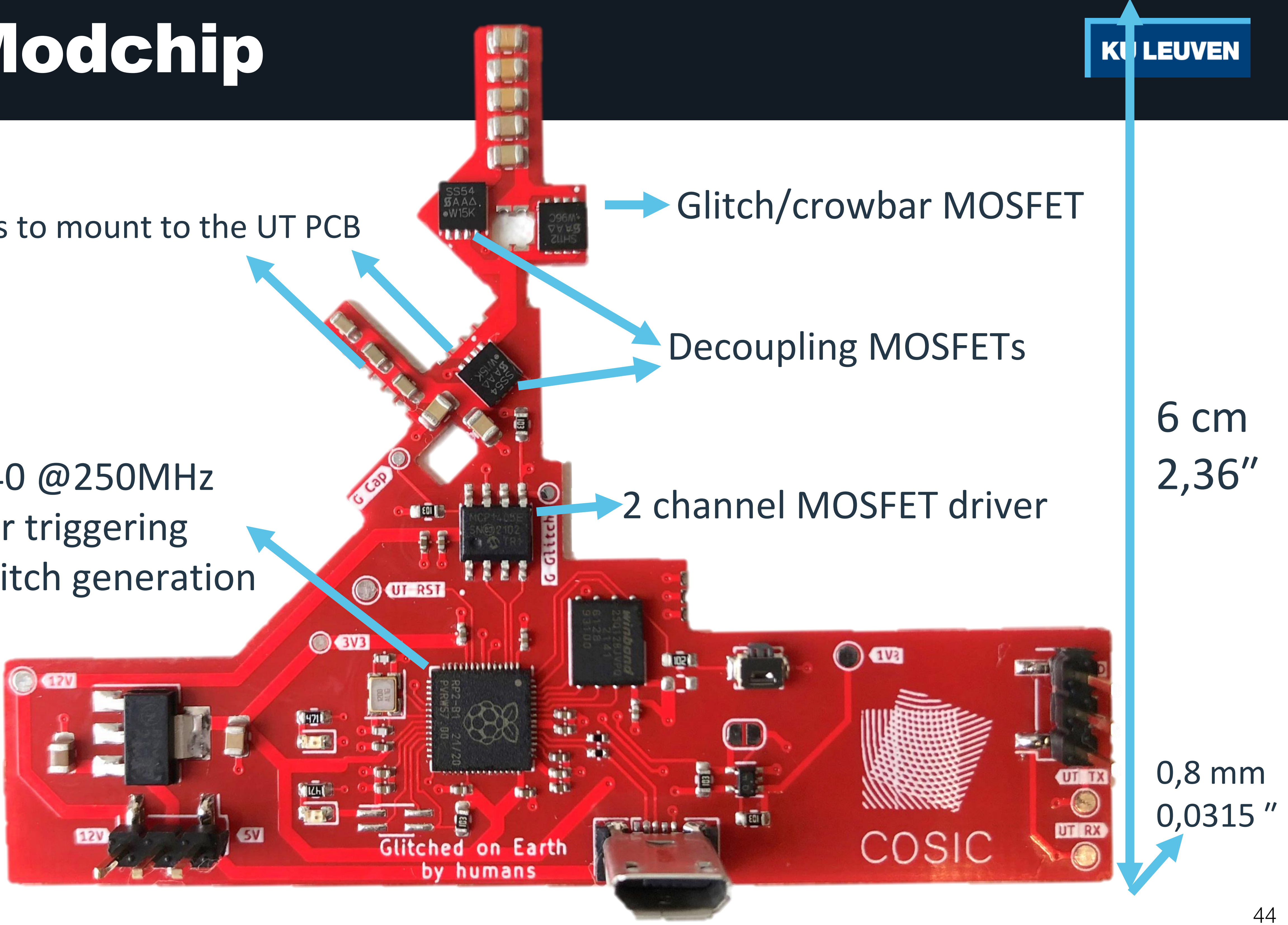
RP2040 @250MHz
PIO for triggering
and glitch generation

2 channel MOSFET driver

6 cm
2,36″

0,8 mm
0,0315 ″

Available on GitHub!

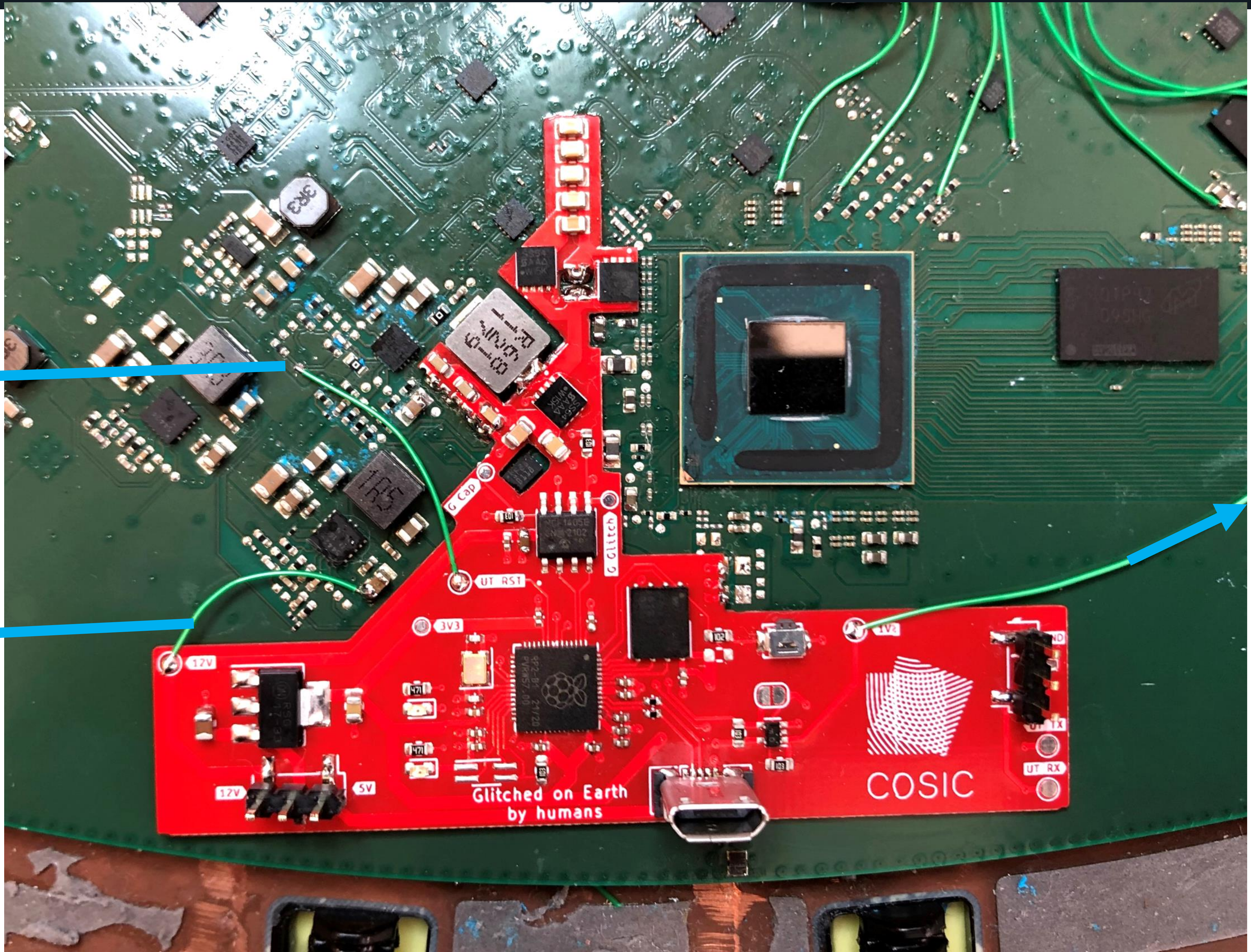


Installed modchip

Core voltage regulator enable pin (for power cycling)

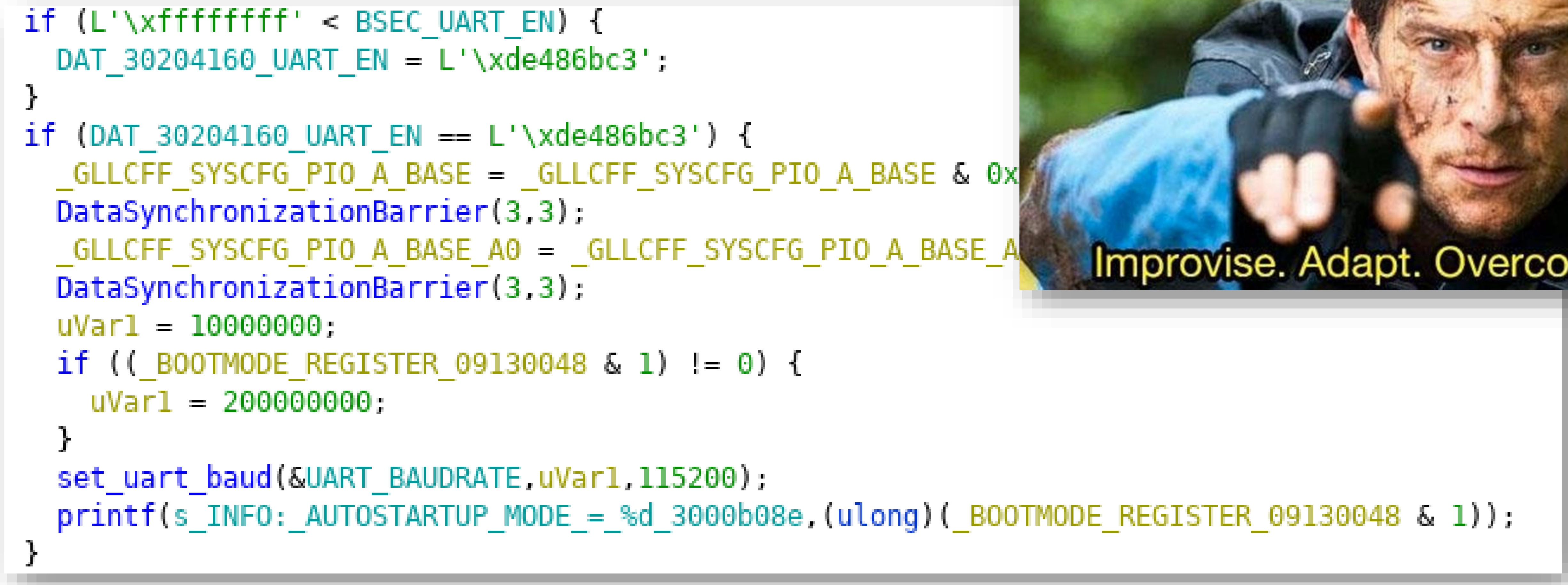
12V for MOSFET drivers and standalone power

1V8 for level shifter



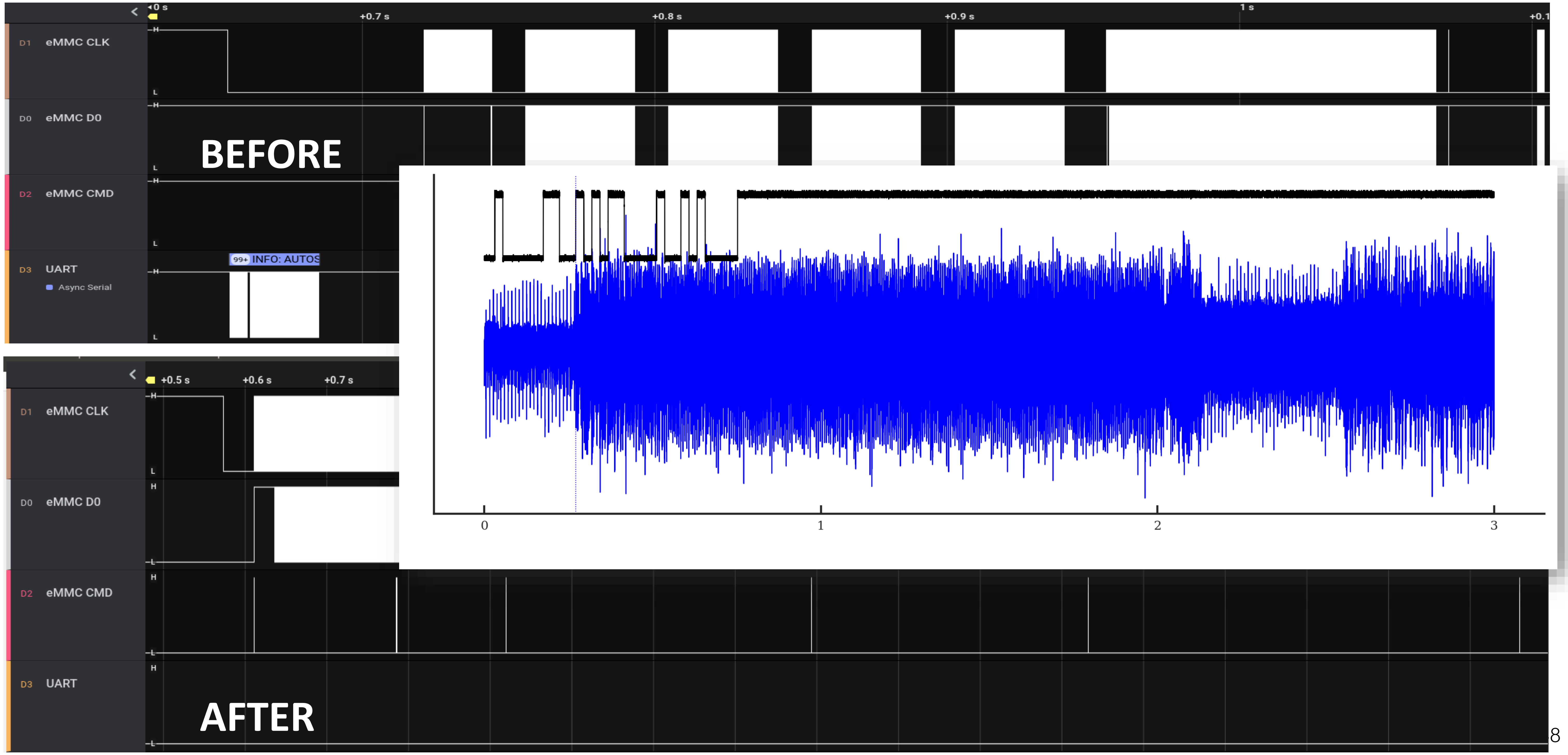
SpaceX strikes back

- I did a firmware update…
- Previously unused eFuse is now blown and disables UART output
- Modchip was designed to trigger on UART



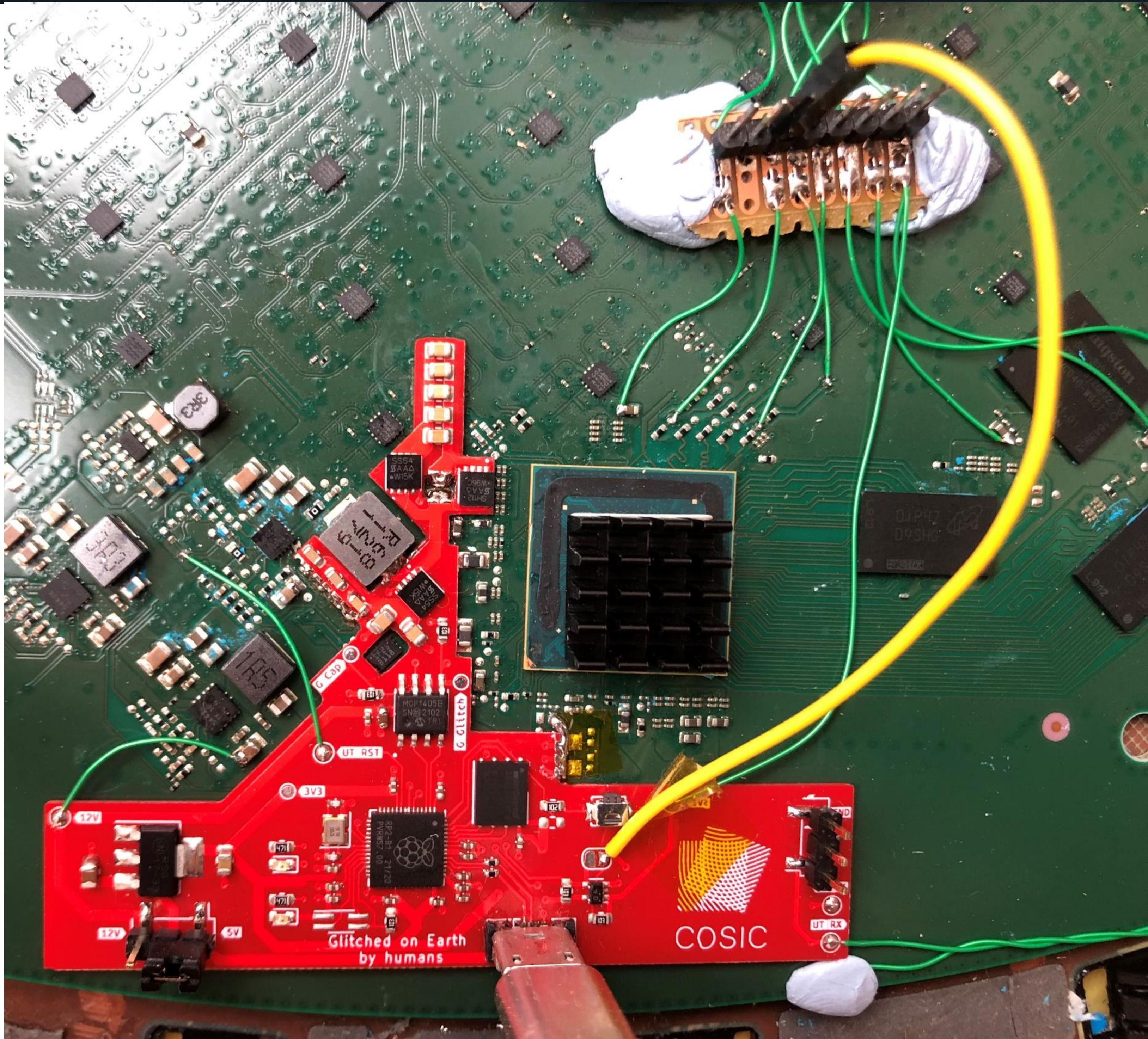
```
if (L'\xffffffff' < BSEC_UART_EN) {
  DAT_30204160_UART_EN = L'\xde486bc3';
}
if (DAT_30204160_UART_EN == L'\xde486bc3') {
  _GLLCFF_SYSCFG_PIO_A_BASE = _GLLCFF_SYSCFG_PIO_A_BASE & 0x
  DataSynchronizationBarrier(3,3);
  _GLLCFF_SYSCFG_PIO_A_BASE_A0 = _GLLCFF_SYSCFG_PIO_A_BASE_A
  DataSynchronizationBarrier(3,3);
  uVar1 = 10000000;
  if ((_BOOTMODE_REGISTER_09130048 & 1) != 0) {
    uVar1 = 200000000;
  }
  set_uart_baud(&UART_BAUDRATE,uVar1,115200);
  printf(s_INFO:_AUTOSTARTUP_MODE_=_%d_3000b08e,(ulong)(_BOOTMODE_REGISTER_09130048 & 1));
}
```

Adapt


BEFORE
AFTER
eMMC CLK
eMMC D0
eMMC CMD
UART
Async Serial
99+ INFO: AUTOS
+0.7 s
+0.8 s
+0.9 s
1 s
+0.5 s
+0.6 s
0
1
2
3

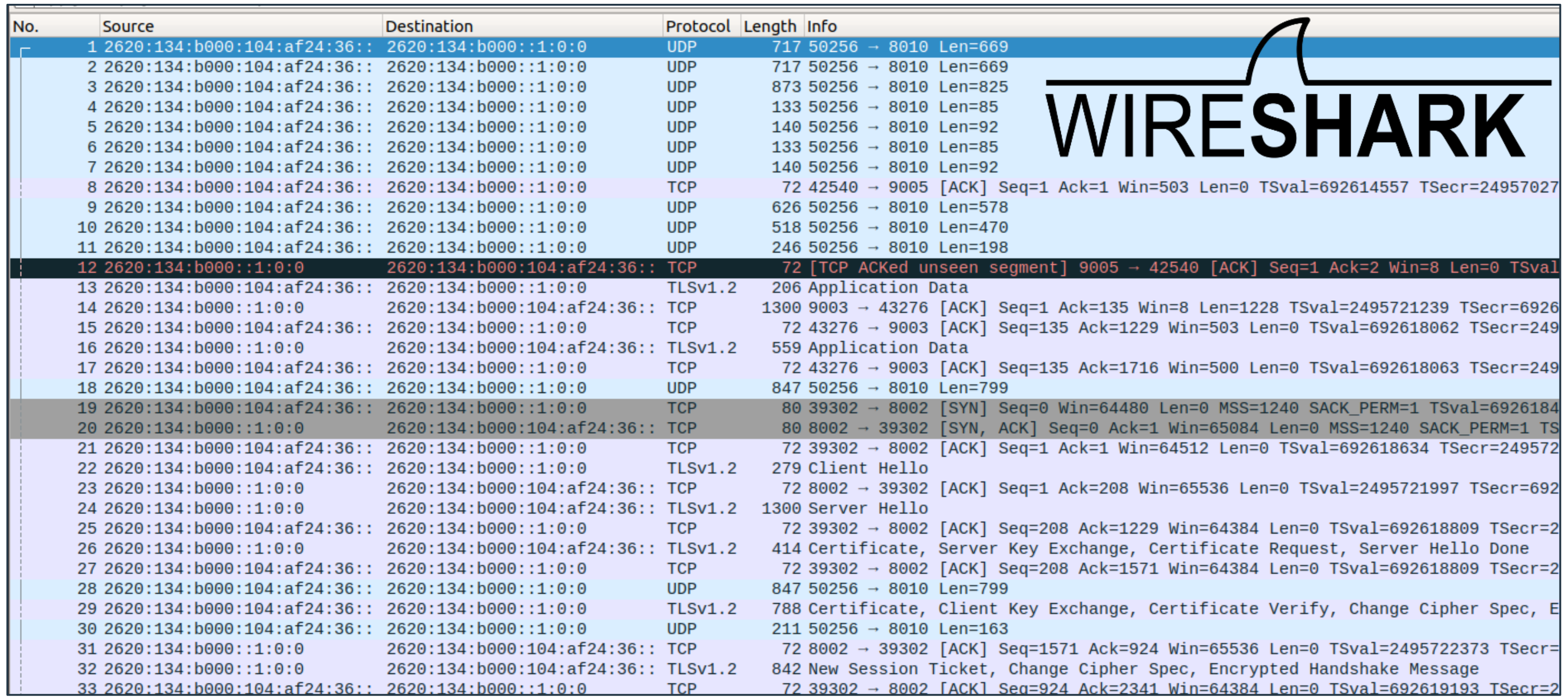
Overcome

- Trigger on eMMC D0 instead of UART
- Modchip could be easily adapted
  - Disconnect UT UART TX
  - Connect to eMMC D0
  - Update glitch parameters from Python
- Alternative: new PCB revision



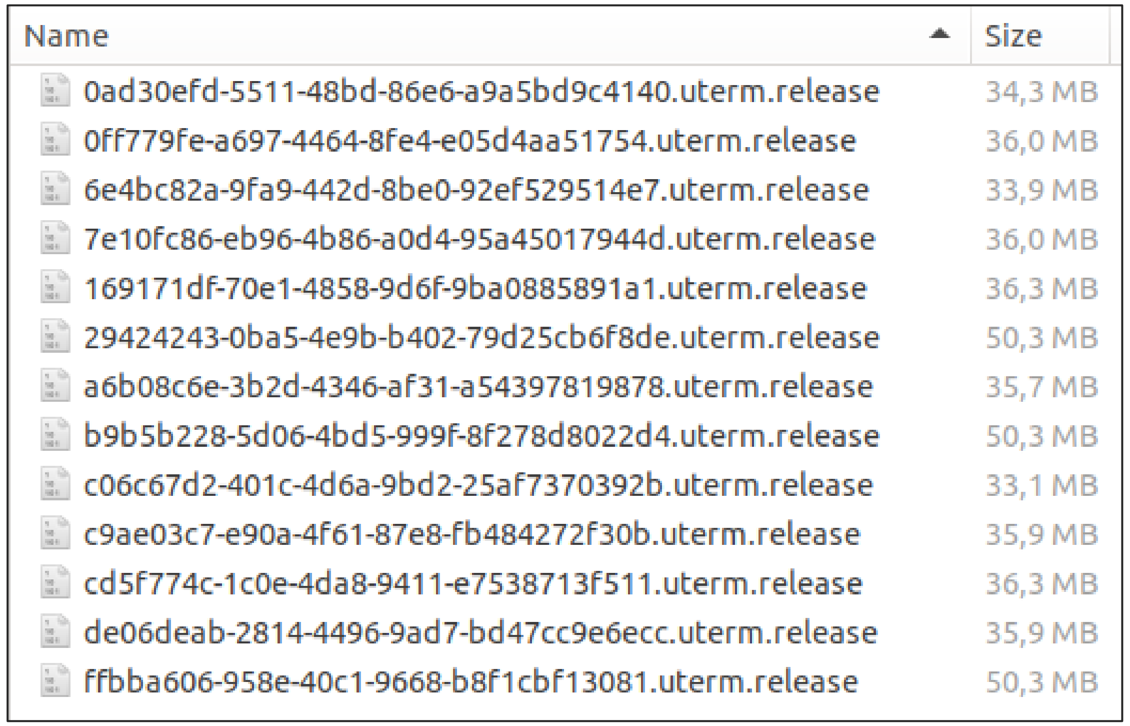
Network exploration

- All interesting communication uses mutually authenticated TLS (STSAFE)
- Added STSAFE support to the tlslite-ng TLS implementation
  - Python script to download the latest firmware updates
- Mostly IPv6 2620:134:b000::1:0:0
  - Open ports (nmap): 8001-8012, 9000, 9003, 9005, 9010, 9011



```
No.  Source                     Destination                Protocol Length Info
   1 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       UDP       717 50256 → 8010 Len=669
   2 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       UDP       717 50256 → 8010 Len=669
   3 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       UDP       873 50256 → 8010 Len=825
   4 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       UDP       133 50256 → 8010 Len=85
   5 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       UDP       140 50256 → 8010 Len=92
   6 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       UDP       133 50256 → 8010 Len=85
   7 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       UDP       140 50256 → 8010 Len=92
   8 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       TCP        72 42540 → 9005 [ACK] Seq=1 Ack=1 Win=503 Len=0 TSval=692614557 TSecr=24957027
   9 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       UDP       626 50256 → 8010 Len=578
  10 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       UDP       518 50256 → 8010 Len=470
  11 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       UDP       246 50256 → 8010 Len=198
  12 2620:134:b000::1:0:0       2620:134:b000:104:af24:36:: TCP        72 [TCP ACKed unseen segment] 9005 → 42540 [ACK] Seq=1 Ack=2 Win=8 Len=0 TSval
  13 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       TLSv1.2   206 Application Data
  14 2620:134:b000::1:0:0       2620:134:b000:104:af24:36:: TCP      1300 9003 → 43276 [ACK] Seq=1 Ack=135 Win=8 Len=1228 TSval=2495721239 TSecr=6926
  15 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       TCP        72 43276 → 9003 [ACK] Seq=135 Ack=1229 Win=503 Len=0 TSval=692618062 TSecr=249
  16 2620:134:b000::1:0:0       2620:134:b000:104:af24:36:: TLSv1.2   559 Application Data
  17 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       TCP        72 43276 → 9003 [ACK] Seq=135 Ack=1716 Win=500 Len=0 TSval=692618063 TSecr=249
  18 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       UDP       847 50256 → 8010 Len=799
  19 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       TCP        80 39302 → 8002 [SYN] Seq=0 Win=64480 Len=0 MSS=1240 SACK_PERM=1 TSval=6926184
  20 2620:134:b000::1:0:0       2620:134:b000:104:af24:36:: TCP        80 8002 → 39302 [SYN, ACK] Seq=0 Ack=1 Win=65084 Len=0 MSS=1240 SACK_PERM=1 TS
  21 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       TCP        72 39302 → 8002 [ACK] Seq=1 Ack=1 Win=64512 Len=0 TSval=692618634 TSecr=249572
  22 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       TLSv1.2   279 Client Hello
  23 2620:134:b000::1:0:0       2620:134:b000:104:af24:36:: TCP        72 8002 → 39302 [ACK] Seq=1 Ack=208 Win=65536 Len=0 TSval=2495721997 TSecr=692
  24 2620:134:b000::1:0:0       2620:134:b000:104:af24:36:: TLSv1.2  1300 Server Hello
  25 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       TCP        72 39302 → 8002 [ACK] Seq=208 Ack=1229 Win=64384 Len=0 TSval=692618809 TSecr=2
  26 2620:134:b000::1:0:0       2620:134:b000:104:af24:36:: TLSv1.2   414 Certificate, Server Key Exchange, Certificate Request, Server Hello Done
  27 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       TCP        72 39302 → 8002 [ACK] Seq=208 Ack=1571 Win=64384 Len=0 TSval=692618809 TSecr=2
  28 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       UDP       847 50256 → 8010 Len=799
  29 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       TLSv1.2   788 Certificate, Client Key Exchange, Certificate Verify, Change Cipher Spec, E
  30 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       UDP       211 50256 → 8010 Len=163
  31 2620:134:b000::1:0:0       2620:134:b000:104:af24:36:: TCP        72 8002 → 39302 [ACK] Seq=1571 Ack=924 Win=65536 Len=0 TSval=2495722373 TSecr=
  32 2620:134:b000::1:0:0       2620:134:b000:104:af24:36:: TLSv1.2   842 New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
  33 2620:134:b000:104:af24:36:: 2620:134:b000::1:0:0       TCP        72 39302 → 8002 [ACK] Seq=924 Ack=2341 Win=64384 Len=0 TSval=692619193 TSecr=2
```

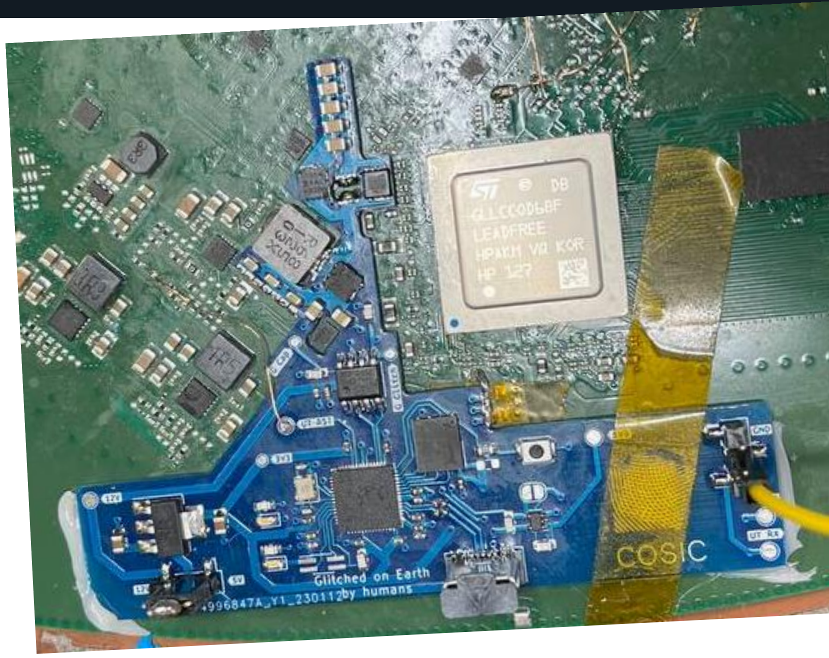
Firmware update archive

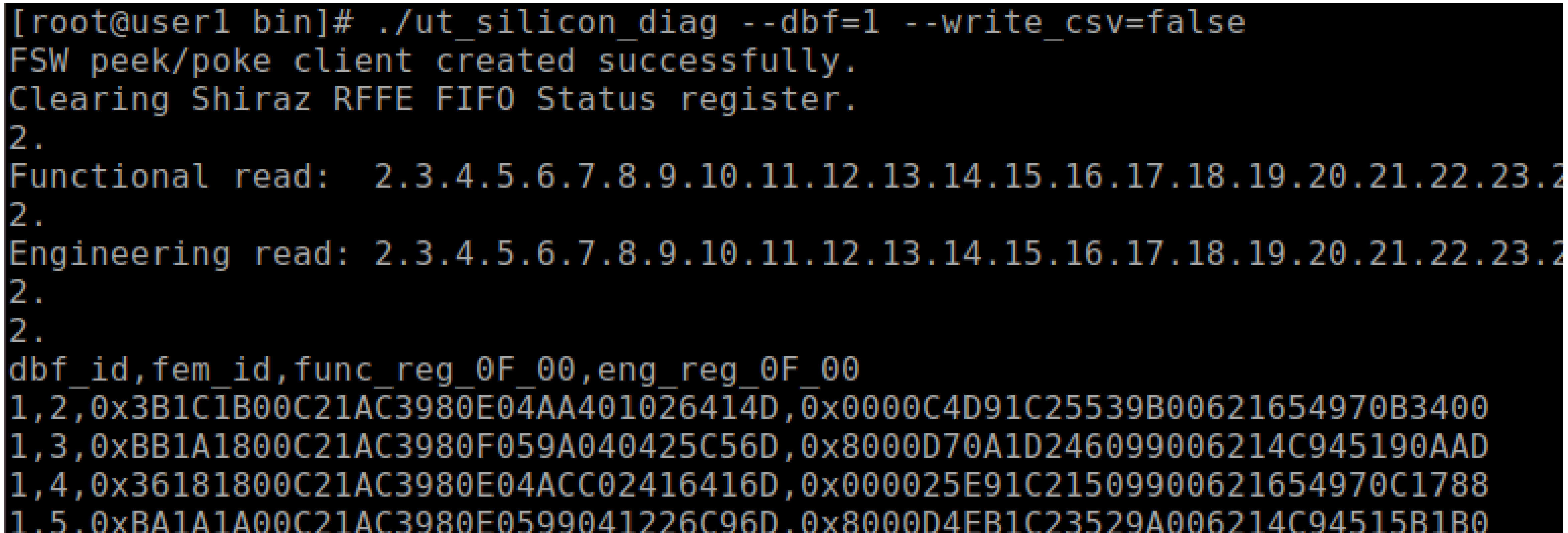


Name	Size
0ad30efd-5511-48bd-86e6-a9a5bd9c4140.uterm.release	34,3 MB
0ff779fe-a697-4464-8fe4-e05d4aa51754.uterm.release	36,0 MB
6e4bc82a-9fa9-442d-8be0-92ef529514e7.uterm.release	33,9 MB
7e10fc86-eb96-4b86-a0d4-95a45017944d.uterm.release	36,0 MB
169171df-70e1-4858-9d6f-9ba0885891a1.uterm.release	36,3 MB
29424243-0ba5-4e9b-b402-79d25cb6f8de.uterm.release	50,3 MB
a6b08c6e-3b2d-4346-af31-a54397819878.uterm.release	35,7 MB
b9b5b228-5d06-4bd5-999f-8f278d8022d4.uterm.release	50,3 MB
c06c67d2-401c-4d6a-9bd2-25af7370392b.uterm.release	33,1 MB
c9ae03c7-e90a-4f61-87e8-fb484272f30b.uterm.release	35,9 MB
cd5f774c-1c0e-4da8-9411-e7538713f511.uterm.release	36,3 MB
de06deab-2814-4496-9ad7-bd47cc9e6ecc.uterm.release	35,9 MB
ffbba606-958e-40c1-9668-b8f1cbf13081.uterm.release	50,3 MB

What’s next?

- You can make your own modchip and use it to:
  - Further explore the network infrastructure
    - Not accessible as a normal user
    - Integrate the STSAFE with GRPC
  - Interact with the Digital BeamFormers and update their firmware
  - Repurpose your terminal?





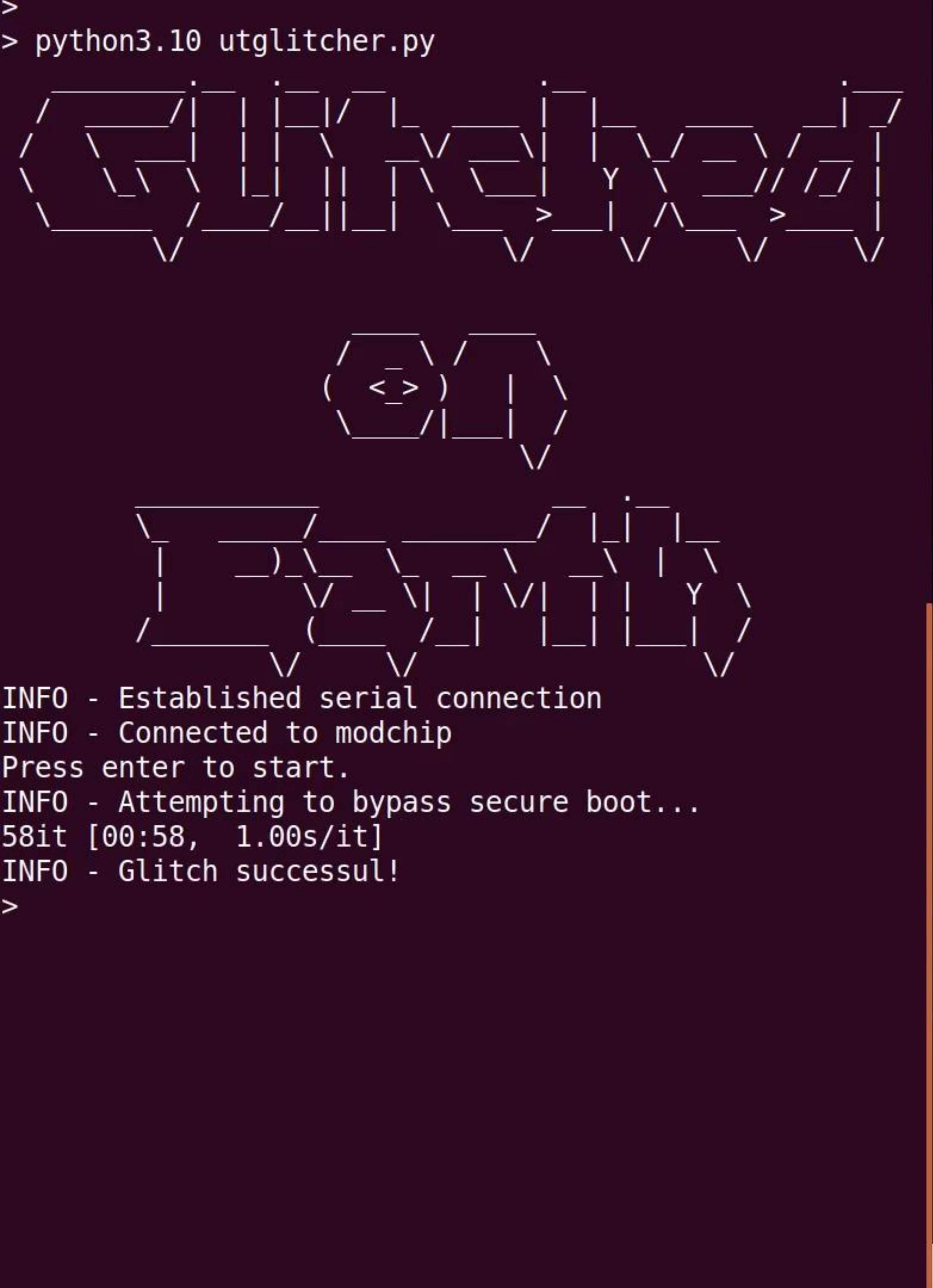
```
[root@user1 bin]# ./ut_silicon_diag --dbf=1 --write_csv=false
FSW peek/poke client created successfully.
Clearing Shiraz RFFE FIFO Status register.
2.
Functional read:  2.3.4.5.6.7.8.9.10.11.12.13.14.15.16.17.18.19.20.21.22.23.2
2.
Engineering read: 2.3.4.5.6.7.8.9.10.11.12.13.14.15.16.17.18.19.20.21.22.23.2
2.
2.
dbf_id,fem_id,func_reg_0F_00,eng_reg_0F_00
1,2,0x3B1C1B00C21AC3980E04AA401026414D,0x0000C4D91C25539B00621654970B3400
1,3,0xBB1A1800C21AC3980F059A040425C56D,0x8000D70A1D246099006214C945190AAD
1,4,0x36181800C21AC3980E04ACC02416416D,0x000025E91C21509900621654970C1788
1,5,0xBA1A1A00C21AC3980F0599041226C96D,0x8000D4EB1C23529A006214C94515B1B0
```

Conclusion

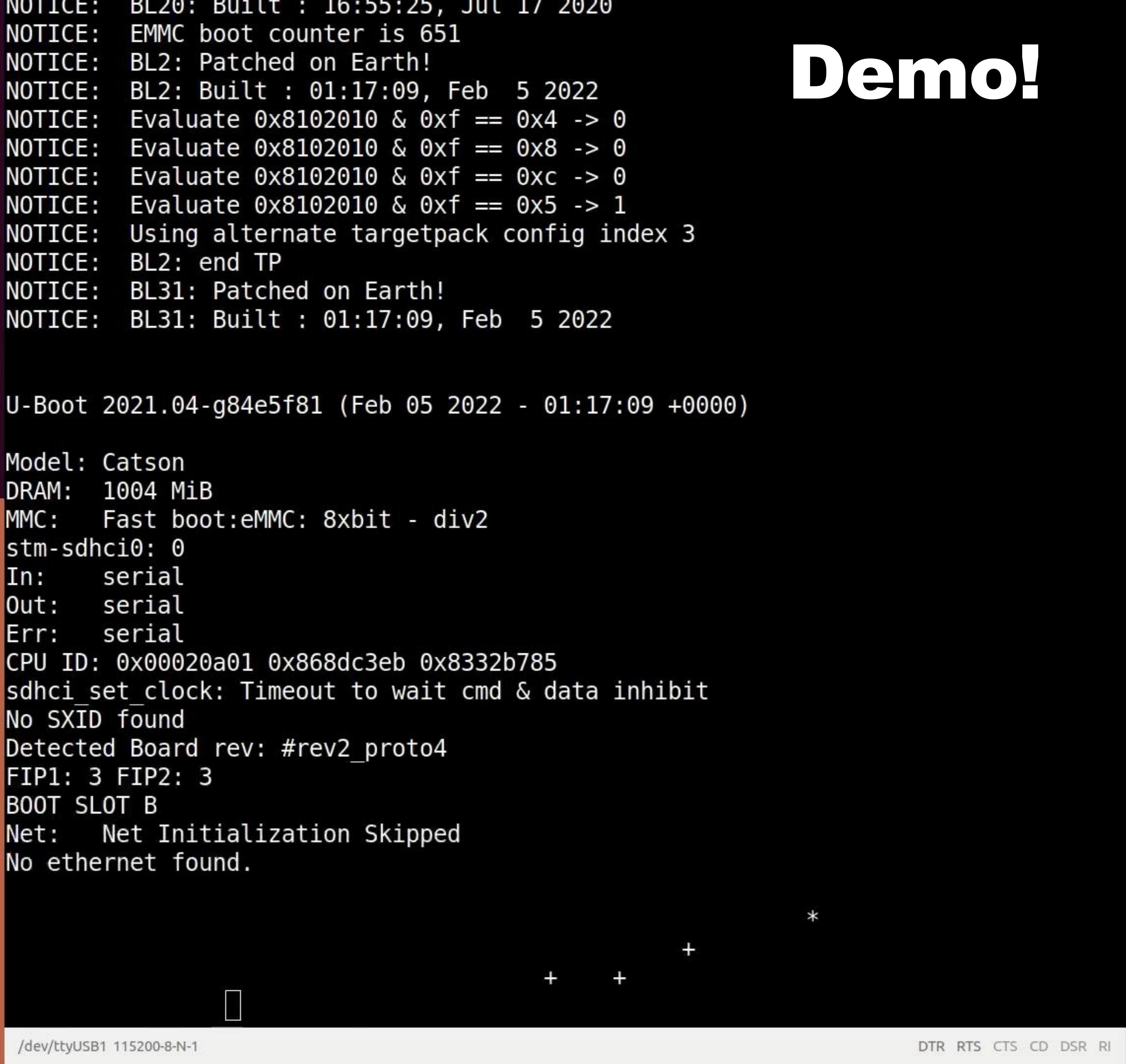
- We can bypass secure boot using voltage fault injection in BL1
  - Quad core Cortex-A53 in a black box scenario
    - no documentation, no open development kits
  - Enabling and disabling of decoupling capacitors
  - Fault injection countermeasures are only as good as the fault model that was used
- This is a well-designed product (from a security standpoint)
  - No obvious (to me) low-hanging fruit
  - In contrast to many other devices getting a root shell was challenging
  - And a root shell does not immediately lead to an attack that scales

- SpaceX PSIRT was very responsive and helpful!
  - https://bugcrowd.com/spacex vulnerabilityreporting@spacex.com
  - https://api.starlink.com/public-files/StarlinkWelcomesSecurityResearchersBringOnTheBugs.pdf

Demo!



```
>
> python3.10 utglitcher.py
  ________.__  .__  __         .__               .___
 /  _____/|  | |__|/  |_  ____ |  |__   ____   __| _/
/   \  ___|  | |  \   __\/ ___\|  |  \_/ __ \ / __ |
\    \_\  \  |_|  ||  | \  \___|   Y  \  ___// /_/ |
 \______  /____/__||__|  \___  >___|  /\___  >____ |
        \/                   \/     \/     \/     \/

                    ____   ____
                   /  _ \ /    \
                  (  <_> )   |  \
                   \____/|___|  /
                              \/
        ___________               __  .__
        \_   _____/____ _______ _/  |_|  |__
         |    __)_\__  \\_  __ \\   __\  |  \
         |        \/ __ \|  | \/ |  | |   Y  \
        /_______  (____  /__|    |__| |___|  /
                \/     \/                  \/
INFO - Established serial connection
INFO - Connected to modchip
Press enter to start.
INFO - Attempting to bypass secure boot...
58it [00:58,  1.00s/it]
INFO - Glitch successul!
>
```



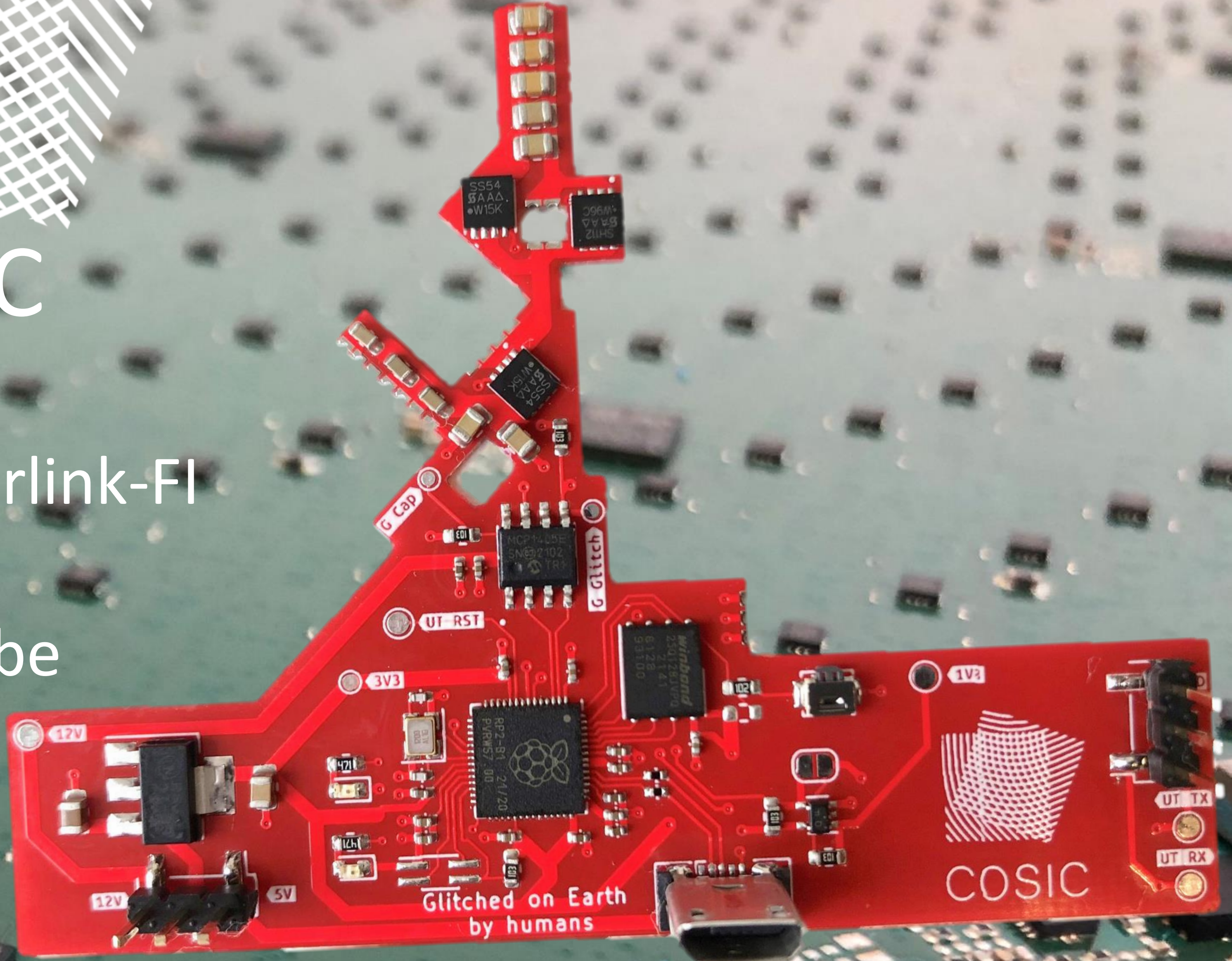
```
NOTICE:  BL20: Built : 16:55:25, Jul 17 2020
NOTICE:  EMMC boot counter is 651
NOTICE:  BL2: Patched on Earth!
NOTICE:  BL2: Built : 01:17:09, Feb  5 2022
NOTICE:  Evaluate 0x8102010 & 0xf == 0x4 -> 0
NOTICE:  Evaluate 0x8102010 & 0xf == 0x8 -> 0
NOTICE:  Evaluate 0x8102010 & 0xf == 0xc -> 0
NOTICE:  Evaluate 0x8102010 & 0xf == 0x5 -> 1
NOTICE:  Using alternate targetpack config index 3
NOTICE:  BL2: end TP
NOTICE:  BL31: Patched on Earth!
NOTICE:  BL31: Built : 01:17:09, Feb  5 2022


U-Boot 2021.04-g84e5f81 (Feb 05 2022 - 01:17:09 +0000)

Model: Catson
DRAM:  1004 MiB
MMC:   Fast boot:eMMC: 8xbit - div2
stm-sdhci0: 0
In:    serial
Out:   serial
Err:   serial
CPU ID: 0x00020a01 0x868dc3eb 0x8332b785
sdhci_set_clock: Timeout to wait cmd & data inhibit
No SXID found
Detected Board rev: #rev2_proto4
FIP1: 3 FIP2: 3
BOOT SLOT B
Net:   Net Initialization Skipped
No ethernet found.

                                                        *
                                              +
                                     +    +

/dev/ttyUSB1 115200-8-N-1                                 DTR RTS CTS CD DSR RI
```

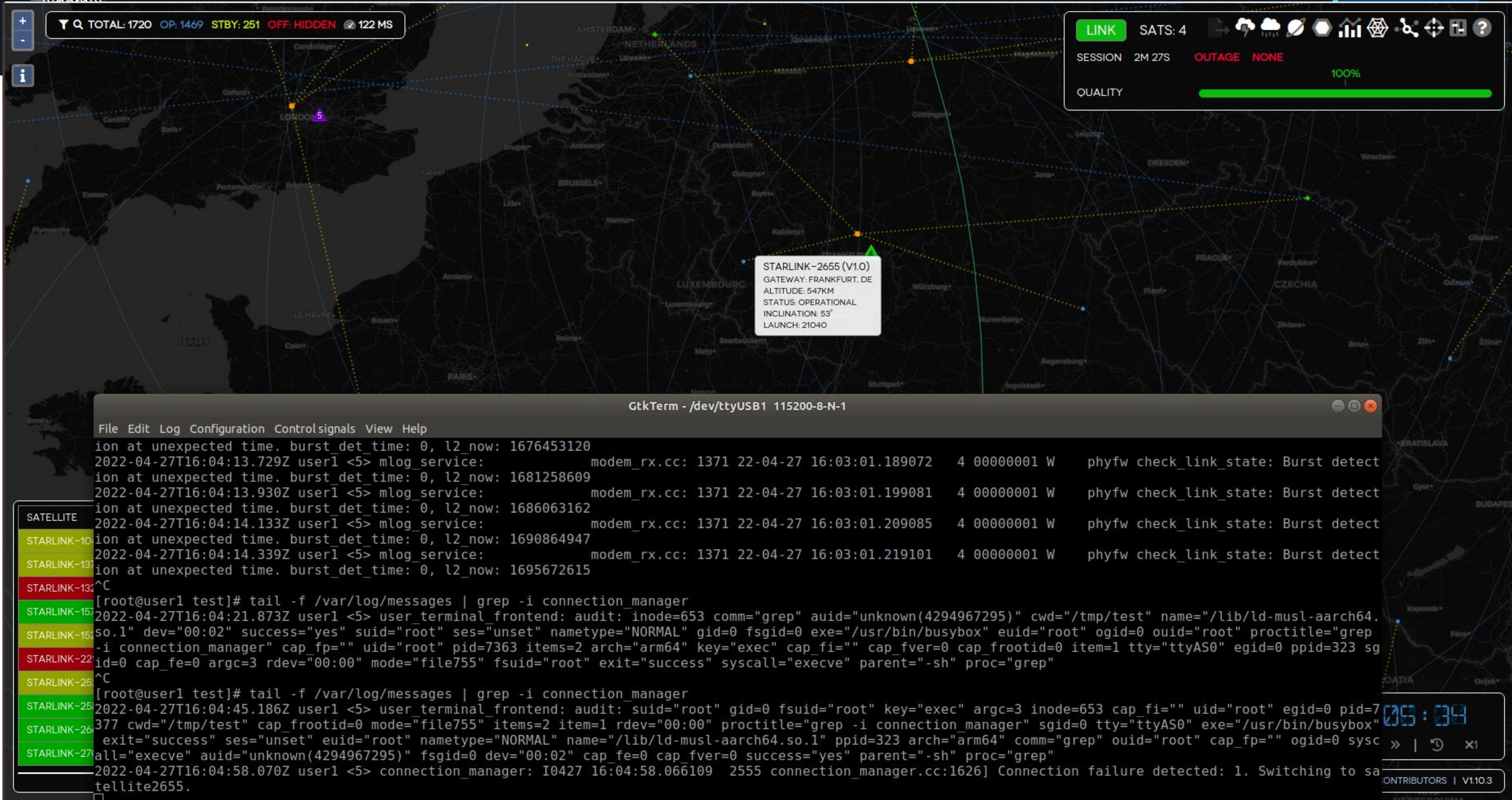
COSIC

github.com/KULeuven-COSIC/Starlink-FI

lennert.wouters@esat.kuleuven.be

@LennertWo





STARLINK-2655 (V1.0)
GATEWAY: FRANKFURT, DE
ALTITUDE: 547KM
STATUS: OPERATIONAL
INCLINATION: 53°
LAUNCH: 21040

```
ion at unexpected time. burst_det_time: 0, l2_now: 1676453120
2022-04-27T16:04:13.729Z user1 <5> mlog_service:      modem_rx.cc: 1371 22-04-27 16:03:01.189072   4 00000001 W    phyfw check_link_state: Burst detect
ion at unexpected time. burst_det_time: 0, l2_now: 1681258609
2022-04-27T16:04:13.930Z user1 <5> mlog_service:      modem_rx.cc: 1371 22-04-27 16:03:01.199081   4 00000001 W    phyfw check_link_state: Burst detect
ion at unexpected time. burst_det_time: 0, l2_now: 1686063162
2022-04-27T16:04:14.133Z user1 <5> mlog_service:      modem_rx.cc: 1371 22-04-27 16:03:01.209085   4 00000001 W    phyfw check_link_state: Burst detect
ion at unexpected time. burst_det_time: 0, l2_now: 1690864947
2022-04-27T16:04:14.339Z user1 <5> mlog_service:      modem_rx.cc: 1371 22-04-27 16:03:01.219101   4 00000001 W    phyfw check_link_state: Burst detect
ion at unexpected time. burst_det_time: 0, l2_now: 1695672615
^C
[root@user1 test]# tail -f /var/log/messages | grep -i connection_manager
2022-04-27T16:04:21.873Z user1 <5> user_terminal_frontend: audit: inode=653 comm="grep" auid="unknown(4294967295)" cwd="/tmp/test" name="/lib/ld-musl-aarch64.
so.1" dev="00:02" success="yes" suid="root" ses="unset" nametype="NORMAL" gid=0 fsgid=0 exe="/usr/bin/busybox" euid="root" ogid=0 ouid="root" proctitle="grep
-i connection_manager" cap_fp="" uid="root" pid=7363 items=2 arch="arm64" key="exec" cap_fi="" cap_fver=0 cap_frootid=0 item=1 tty="ttyAS0" egid=0 ppid=323 sg
id=0 cap_fe=0 argc=3 rdev="00:00" mode="file755" fsuid="root" exit="success" syscall="execve" parent="-sh" proc="grep"
^C
[root@user1 test]# tail -f /var/log/messages | grep -i connection_manager
2022-04-27T16:04:45.186Z user1 <5> user_terminal_frontend: audit: suid="root" gid=0 fsuid="root" key="exec" argc=3 inode=653 cap_fi="" uid="root" egid=0 pid=7
377 cwd="/tmp/test" cap_frootid=0 mode="file755" items=2 item=1 rdev="00:00" proctitle="grep -i connection_manager" sgid=0 tty="ttyAS0" exe="/usr/bin/busybox"
 exit="success" ses="unset" euid="root" nametype="NORMAL" name="/lib/ld-musl-aarch64.so.1" ppid=323 arch="arm64" comm="grep" ouid="root" cap_fp="" ogid=0 sysc
all="execve" auid="unknown(4294967295)" fsgid=0 dev="00:02" cap_fe=0 cap_fver=0 success="yes" parent="-sh" proc="grep"
2022-04-27T16:04:58.070Z user1 <5> connection_manager: I0427 16:04:58.066109  2555 connection_manager.cc:1626] Connection failure detected: 1. Switching to sa
tellite2655.
```