# Breaking Espressif's ESP32 V3: Program Counter Control with Computed Values using Fault Injection

Jeroen Delvaux, Cristofaro Mune, Mario Romero, Niek Timmers

all contributed equally

Technology Innovation Institute

raelize

Usenix WOOT 2024, Philadelphia, USA

# Our Target: ESP32 V3

**ESP32 V1 [2016]**

- The ESP32 is a widely popular *system-on-chip* (SoC) with WiFi and Bluetooth.
- The CPU of the V1 chip implements the Xtensa *instruction set* (not RISC-V).
- Security features: *Secure Boot* and *Flash Encryption*.
- Left, a development board from Espressif is shown.

Broken by multiple *fault injection* (FI) attacks from multiple research teams.

**ESP32 V3 [2020]**

Hardened:

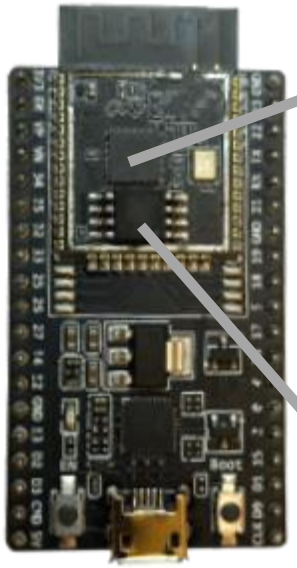
- The ROM code is protected against FI by introducing redundancies
- Cryptographic overhaul of secure boot: from symmetric-key crypto to public-key crypto

We defeat all security features with a single electromagnetic (EM) glitch. Findings were responsibly disclosed:

- Security Advisory AR2023-005
- CVE-2023-35818.

# Secure Boot and Flash Encryption

Original
*dev board*



Metal shield
removed

ESP32 V3 SoC

ROM

SRAM

Fuses

- AES key
- Hash of RSA public key

Crypto accelerators

Xtensa CPU

Flash chip

Encrypted by AES

- bootloader (and application) code
- RSA public key & signature

Chain of trust:

ROM code

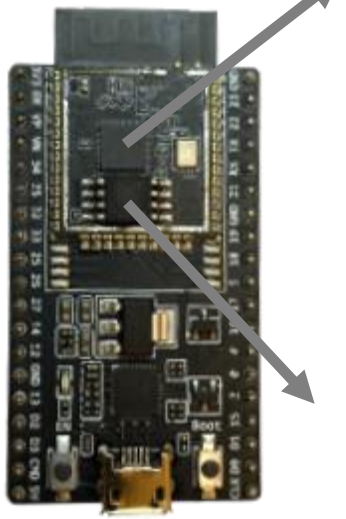
verifies

Bootloader

verifies

Application

# Program Counter Control using EM-FI



ESP32 V3 SoC

ROM

SRAM

Fuses

Crypto accelerators

Xtensa CPU

General-purpose (GP) registers

Program Counter (PC)

Flash

Encrypted

bootloader

signature block

**(2) Physical vulnerability:**
EM-FI corrupts instructions and transfers the pointer to the PC.

**(1) Mathematical vulnerability:**
Modify ciphertext to insert an arbitrary 32-bit plaintext pointer into GP registers.

Chain of trust:

ROM code

~~verifies~~

PC control happens here and cannot be precluded by a software patch

Bootloader

verifies

Application

# Serial (UART)

Valid bootloader, printing “Hello, World!”

```
ets Jul 29 2019 12:21:46
rst:0x1 (POWERON_RESET),boot:0x13
(SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,
hd_drv:0x00,wp_drv:0x00
mode:2, clock div:2
secure boot v2 enabled
secure boot verification succeeded
load:0x3fff0020 len:0xc8c
load:0x40078000 len:0x2020
load:0x40080400 len:0xeac
entry 0x40080640
I (41) boot: ESP-IDF v5.0.1-397-g3050ea656f 2nd
stage bootloader
I (41) boot: compile time 16:51:07
I (41) boot: chip revision: v3.0
I (45) boot.esp32: SPI Speed      : 40MHz
I (50) boot.esp32: SPI Mode       : DIO
I (54) boot.esp32: SPI Flash Size : 2MB
I (59) boot: Enabling RNG early entropy source...
Hello, World!
```

We manipulate ciphertext such that a CRC32 checksum over the signature block fails and prints our pointer

```
ets Jul 29 2019 12:21:46
rst:0x1 (POWERON_RESET),boot:0x13
(SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,
hd_drv:0x00,wp_drv:0x00
mode:2, clock div:2
secure boot v2 enabled
Sig block 0 invalid: Stored CRC 0xbaaeaf78
calculated 0xdeadbeef
secure boot verification failed
```

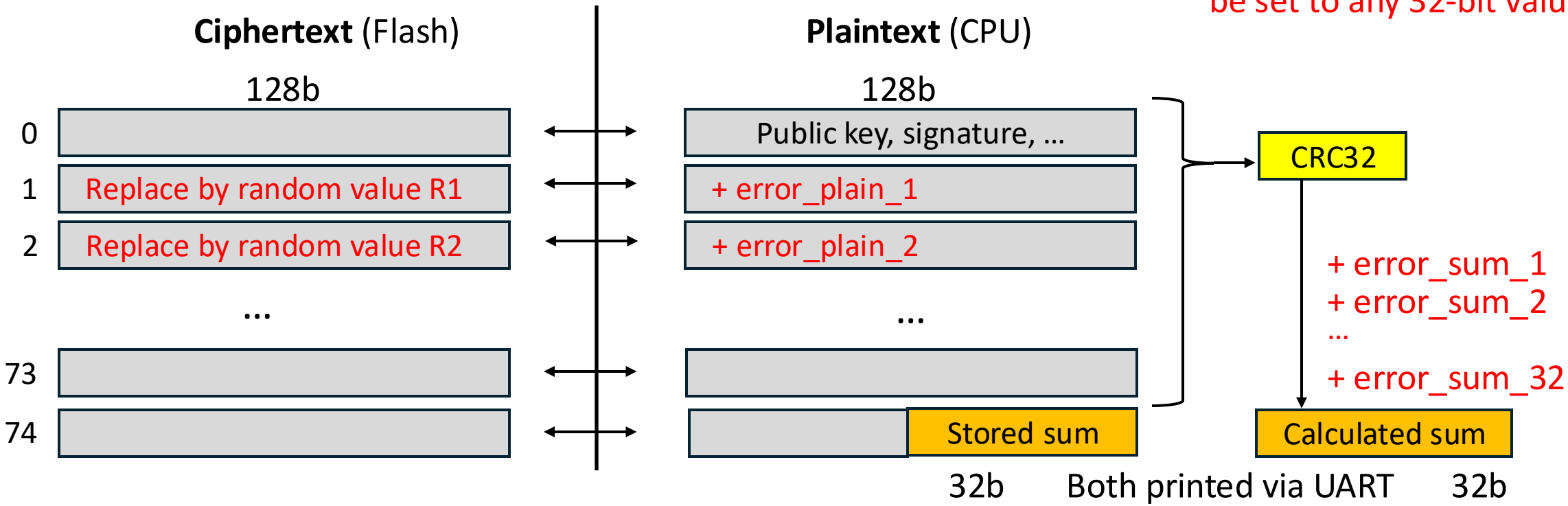
| Pointer | Practical use |
|---|---|
| **0xdeadbeef** | None; hexpeak |
| **0x80006864** | A print statement in ROM, kindly letting us know that PC control is successful |
| **0x80008ceb** | ROM entry point for *Download Mode*. Here, the entire Flash contents can be decrypted. |

# Pointer Insertion via CRC32 Checksum

**Enabling factors:**

- CRC32 is an affine/linear function
- Checksums are printed via UART
- AES operates on 128-bit blocks independently

By solving a system of 32 linear equations, the calculated checksum can be set to any 32-bit value



# Pointer Transfer to Program Counter

Espressif published the ROM code as an *.elf file, which allows for reverse engineering in Ghidra.

**ets_secure_boot_verify_signature**

```
0x40065474  movi     a12, 0x4ac
0x40065477  movi.n   a10, 0x0
0x40065479  mov.n    a11, a6
0x4006547b  movi     a2, 0x4ac
0x4006547e  call8    crc32_le
0x40065481  add.n    a2, a6, a2
0x40065483  l32i.n   a12, a2, 0x0
0x40065485  mov      a13, a10
0x40065488  beq      a10, a12, 0x40065498
0x4006548b  l32r     a10, 0x40065428
0x4006548e  mov      a11, a7
0x40065491  call8    ets_printf
0x40065494  j        0x40065565
```

**crc32_le**

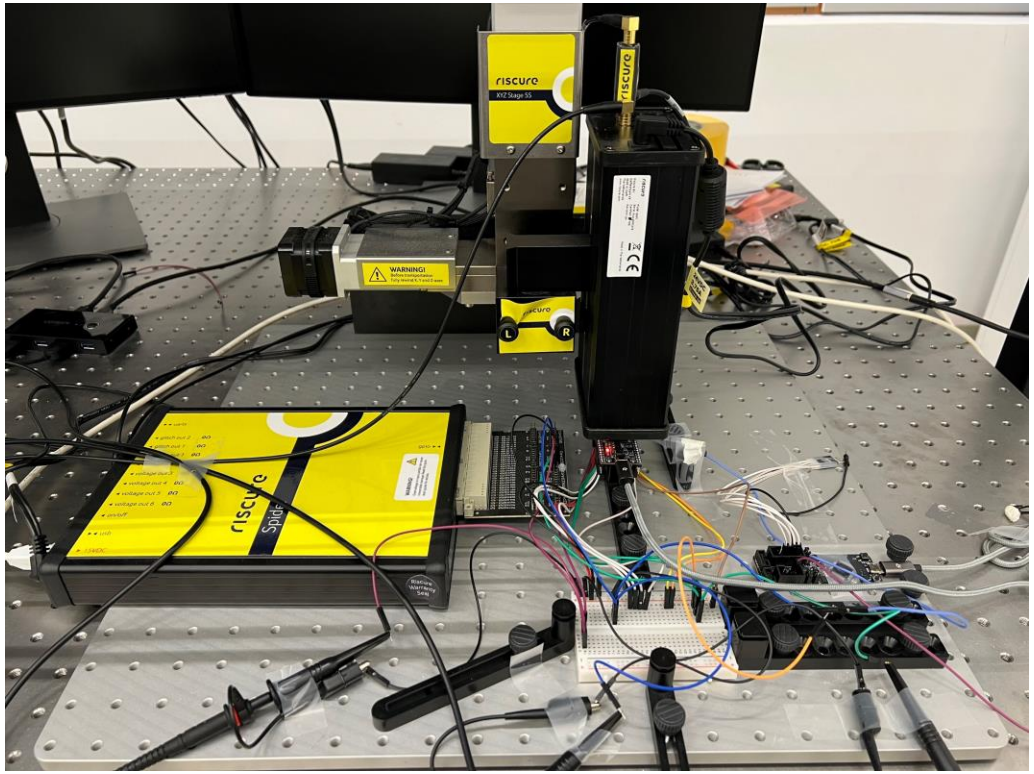
```
0x4005cfec  entry    a1, 0x20
0x4005cfef  movi.n   a8, 0xff
0x4005cff1  xor      a2, a8, a2
0x4005cff4  l32r     a9, 0x4005cfe8
0x4005cff7  movi.n   a8, 0x0
0x4005cff9  j        0x4005d014
0x4005cffc  add.n    a10, a3, a8
0x4005cffe  l8ui     a10, a10, 0x0
0x4005d001  addi.n   a8, a8, 0x1
0x4005d003  xor      a10, a10, a2
0x4005d006  extui    a10, a10, 0x0, 0x8
0x4005d009  addx4    a10, a10, a9
0x4005d00c  l32i.n   a10, a10, 0x0
0x4005d00e  srli     a2, a2, 0x8
0x4005d011  xor      a2, a10, a2
0x4005d014  bne      a8, a4, 0x4005cffc
0x4005d017  movi.n   a3, 0xff
0x4005d019  xor      a2, a3, a2
0x4005d01c  retw.n
```

**ets_printf**

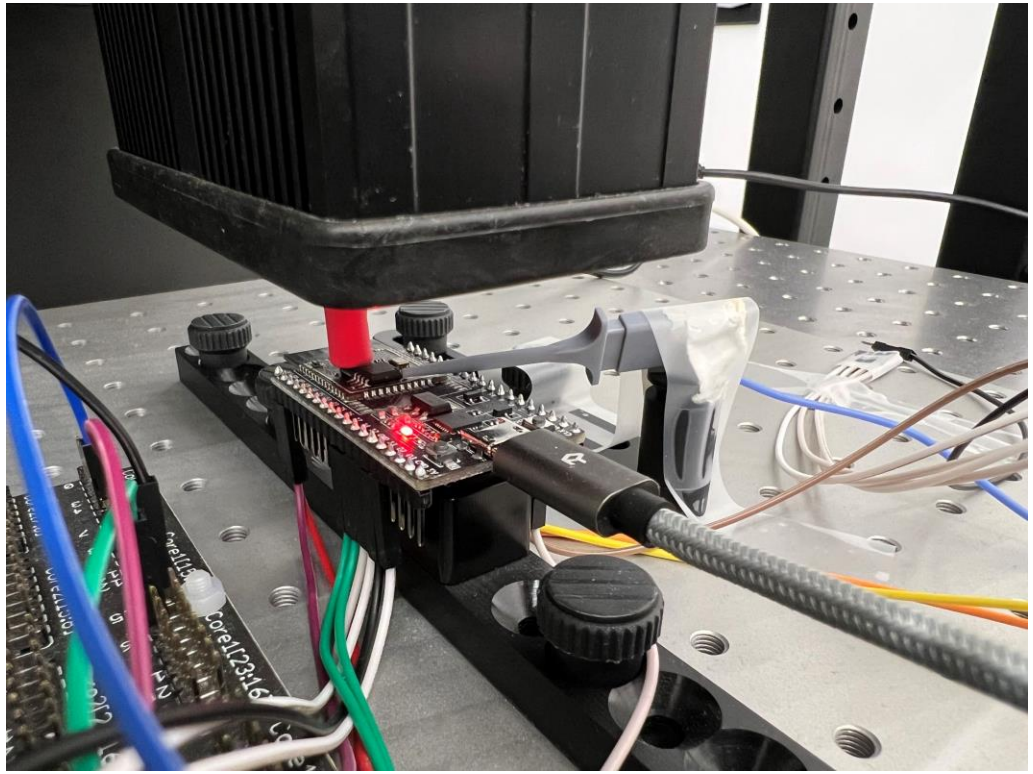
```
Sig block 0 invalid: Stored CRC 0xbaaeaf78
calculated 0xdeadbeef
```

- The pointer passes through several GP registers.
- EM-FI corrupts CPU instructions, thereby potentially transferring the pointer to the PC register.
- Overwriting register a0, which stores a function's *return address*, is a likely pathway.

# Riscure EM-FI Setup




XYZ stage, Spider, breadboard



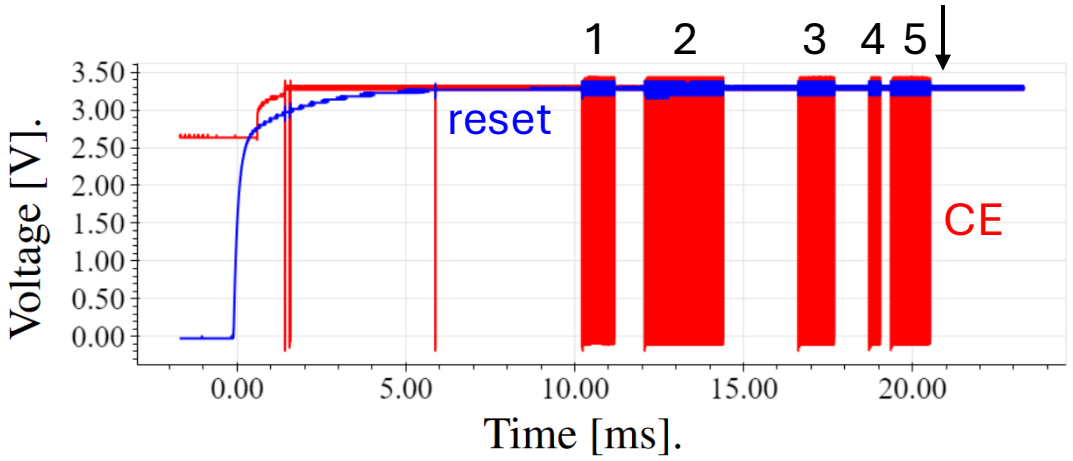
ESP32 V3, probe

- The surface of the ESP32 V3 chip is partially blocked by the Flash chip.
- Displacing the Flash chip is not needed to succeed.

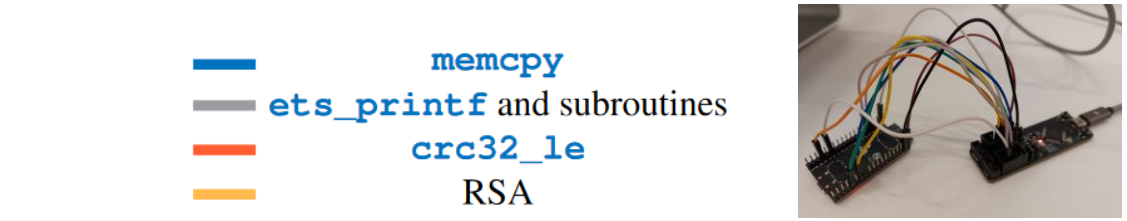
# Rough Timing of the Glitch

Timing reference: *chip-enable* (CE) signal of the Flash chip:

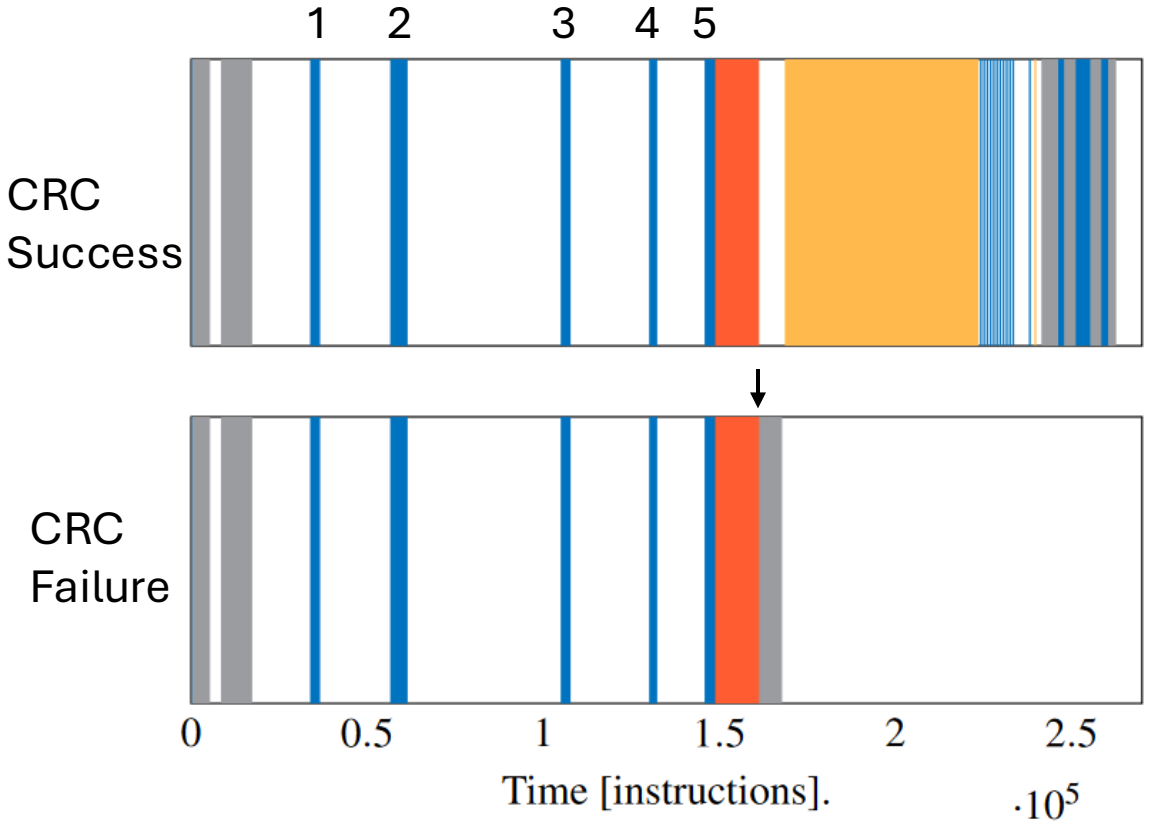
- Five large blocks where data is copied.
- The glitch is injected shortly after block 5.




GNU Debugger (GDB) execution trace





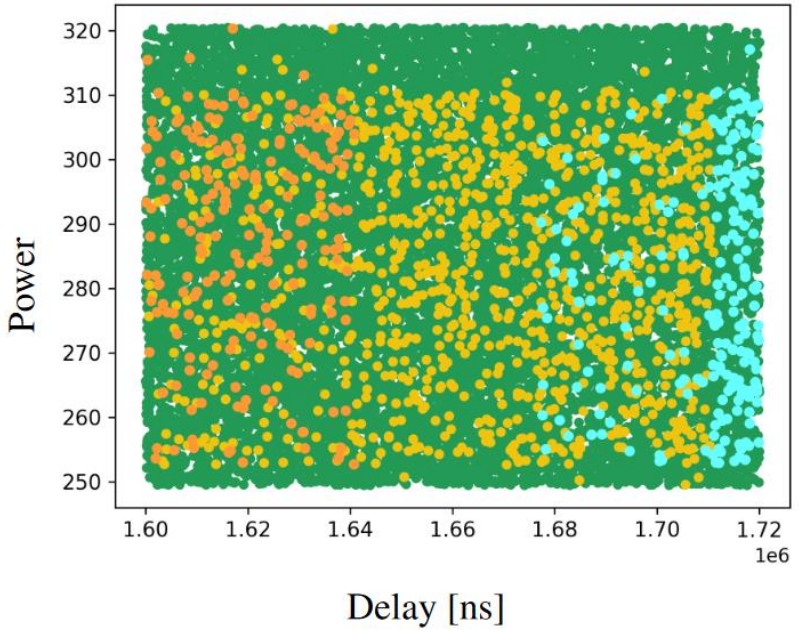
# Fine Timing of the Glitch

Using FI as a virtual oscilloscope.

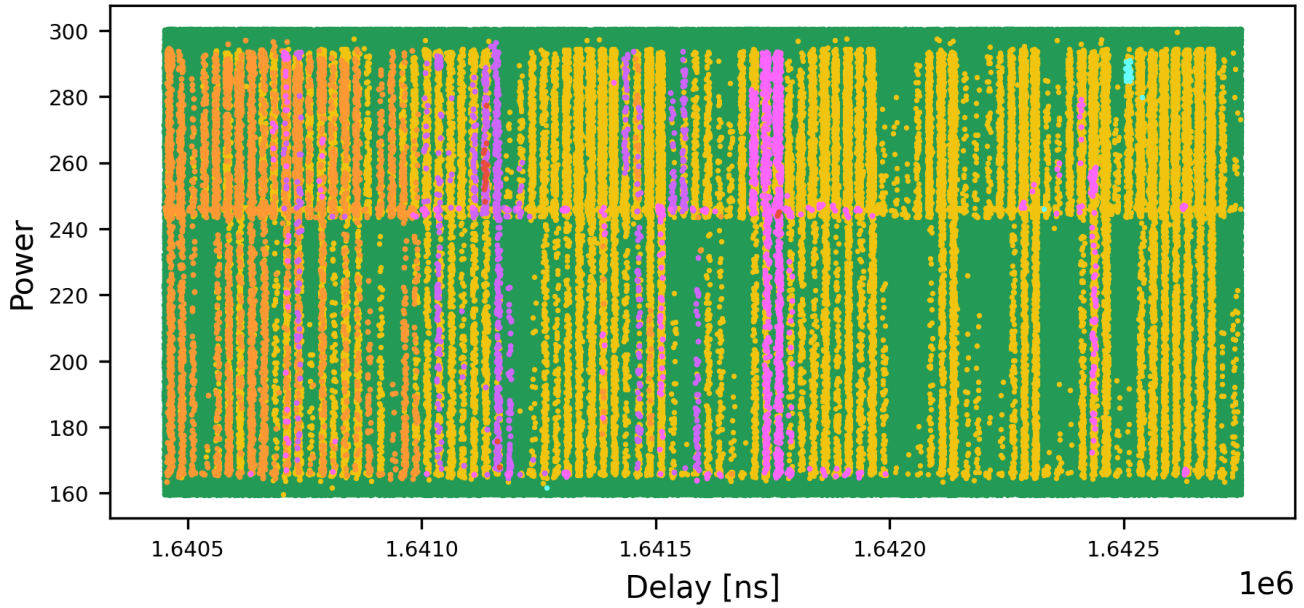
```
Sig block 0 invalid: Stored CRC 0xbaaeaf78
calculated 0xdeadbeef
```

- Nominal
- Crash
- Calculated sum is altered
- Stored sum is altered
- Calculated and stored sums are both altered
- Deformed checksum string
- PC control

**Larger window**



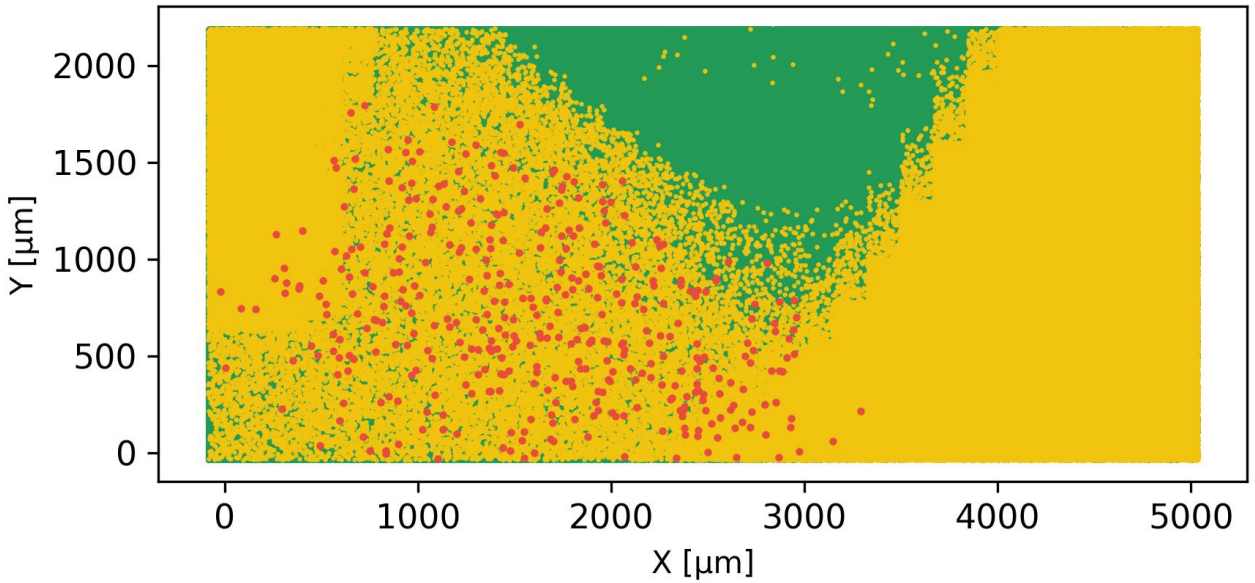

**Smaller window**



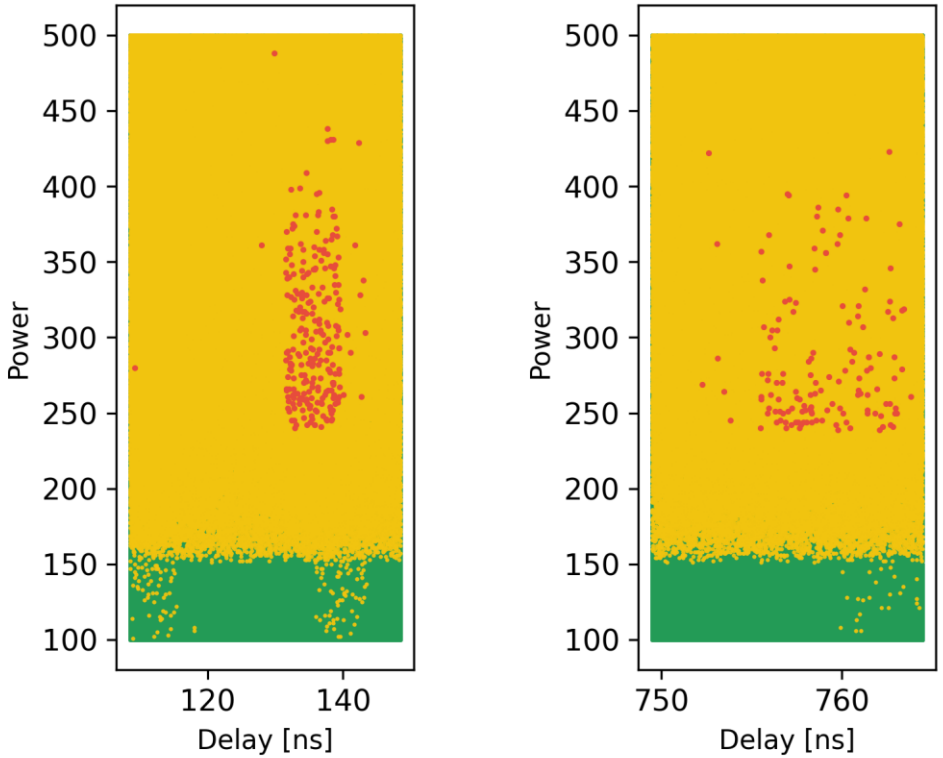
# Scan Results

- nominal
- crash
- PC control

A large fraction of the ESP chip surface is vulnerable.




At least two places in the ROM code are vulnerable.




Speed: 3.4 glitches per second. A few days sufficed to find the first ‘red’ dot.

# Download Mode

- Download Mode is used by developers and is disabled in the field by burning a fuse.
- Nevertheless, we enter Download Mode by taking PC Control.
- Upon entering, we send and receive packets via UART to decrypt the Flash contents.

Request: c0000a0400000000**00010403f**c0 Read 32 bits at virtual address 0x3f401000.

Reply: c0010a0400**e9030210**0000000c0 The requested 32-bit word is 0x100203e9.

**Ciphertext** (Flash)

```
00000000 BB C3 FC 39 C1 52 A1 1B 05 D8 E9 FF A2 4E D3 64 ...9.R.......N.d
00000010 7C 55 95 FC DC 5C AA BB AC 81 38 A1 0F 99 62 42 |U...\....8...bB
00000020 98 D1 9C 13 66 1C 49 D1 E4 C4 42 6F D9 76 24 55 ....f.I...Bo.v$U
00000030 DD 4A C4 ED FB 01 05 18 29 02 4A 7A F4 01 4E 52 .J.......).Jz..NR
00000040 C1 2C B9 02 77 6F DE 4B 72 24 1A DB 2D A9 1D 3E .,..wo.Kr$..-..>
00000005 39 E1 0D BB A3 6F BA B1 DA E5 02 A0 27 76 00 64 9....o......'v.d
```

**Plaintext** (CPU)

```
00000000 E9 03 02 10 3C 06 08 40 EE 00 00 00 00 00 03 00 ....<..@........
00000010 00 FF FF 00 00 00 00 01 20 00 FF 3F C4 0C 00 00 ........ ..?....
00000020 FF FF FF FF 28 50 04 00 FF AC 00 00 01 00 00 00 ....(P..........
00000030 00 F0 F5 3F 00 00 00 00 04 00 00 00 05 00 00 00 ...?............
00000040 06 00 00 00 07 00 00 00 41 73 73 65 72 74 20 66 ........Assert f
00000005 61 69 6C 65 64 20 69 6E 20 25 73 2C 20 25 73 3A ailed in %s, %s:
```

# Concluding Remarks

- Main lesson: avoid printing plaintext data, or results from a plaintext computation.
- We thank Espressif for establishing a smooth vulnerability disclosure process.
- Espressif indicated that the attack does not apply to ESP32-S2, ESP32-C3, ESP32-S3, and future chips.
- Taking PC control by inserting pointers in a computation is a generic methodology for attacks, possibly affecting manufacturers other than Espressif.
- Thank you for attending!