

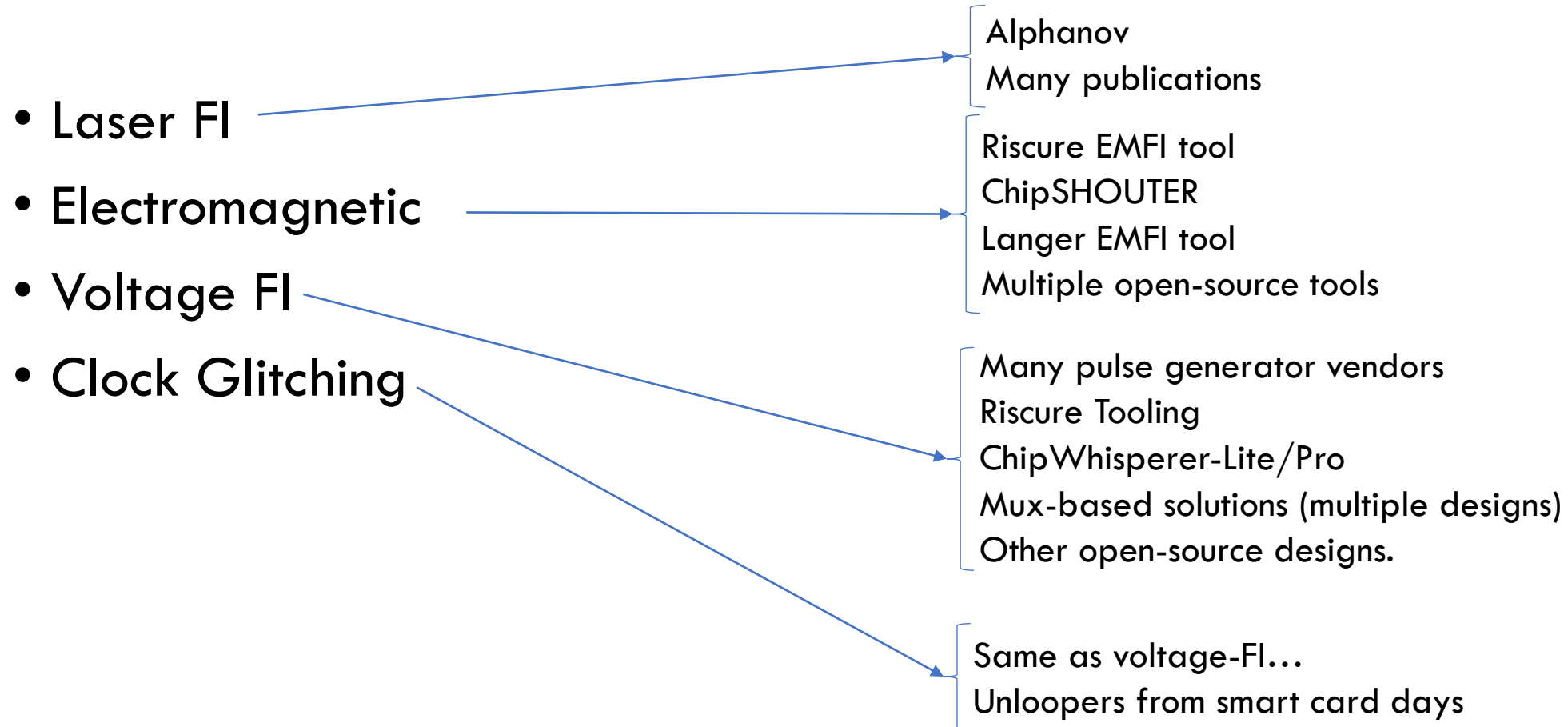
Low-Cost Body Biasing Injection (BBI) Attacks on WLCSP Devices

Colin O'Flynn

Dalhousie University / NewAE

<https://github.com/newaetech/chipjabber-basicbbi>

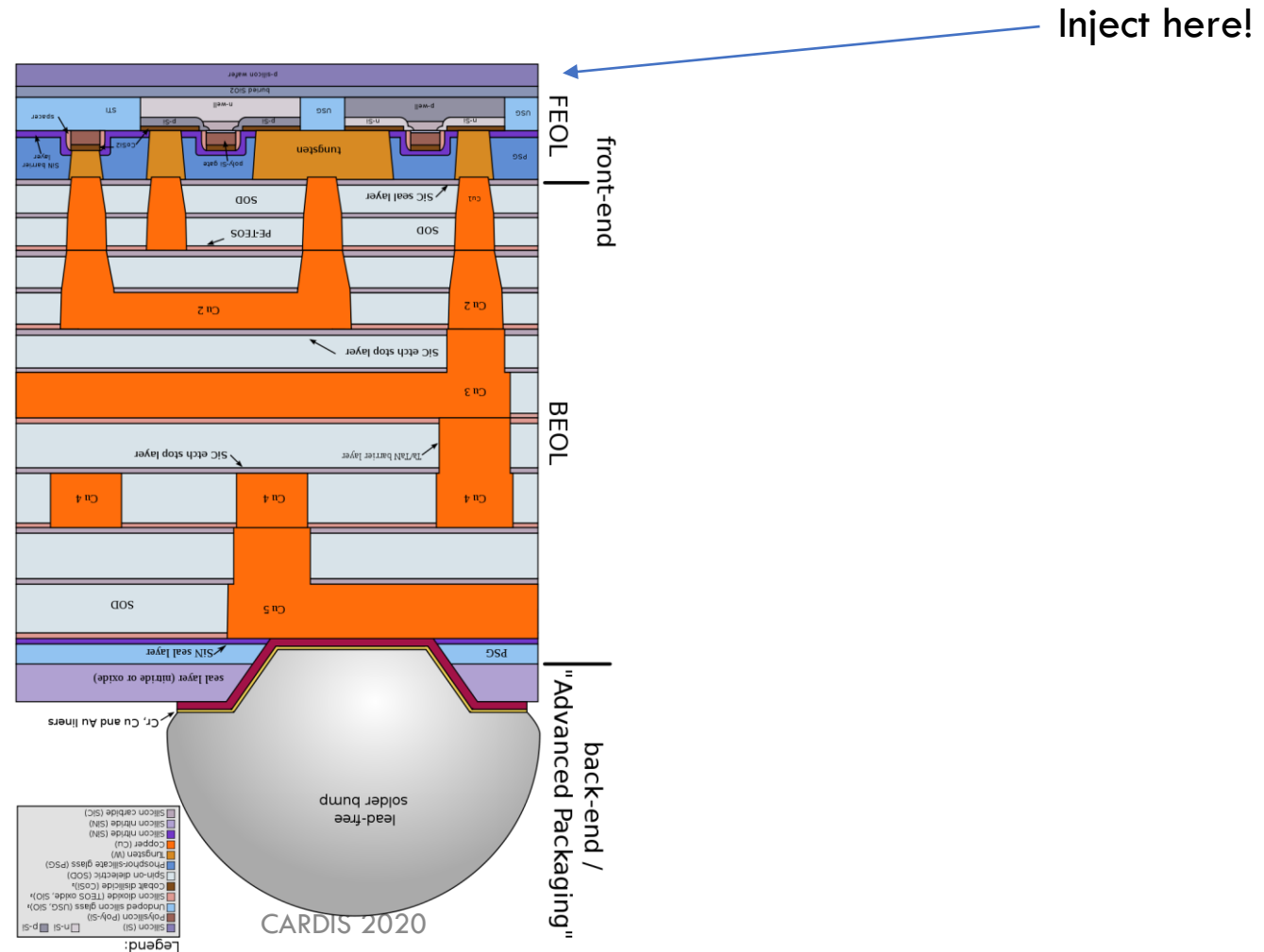
Popular Ways to Inject Faults



What happened to Body Biasing Injection (BBI)?

- Philippe Maurine. **Techniques for EM Fault Injection: Equipments and Experimental Results**. FDTC 2012: Workshop on Fault Diagnosis and Tolerance in Cryptography. 9 Sept. 2012.
- Philippe Maurine, Karim Tobich, Thomas Ordas, Pierre Yvan Liardet. **Yet Another Fault Injection Technique: by Forward Body Biasing Injection**. YACC'2012: Yet Another Conference on Cryptography, Sep 2012, Porquerolles Island, France.
- Karim Tobich, Philippe Maurine, Pierre Yvan Liardet, Mathieu Lisart, Thomas Ordas. **Voltage Spikes on the Substrate to Obtain Timing Faults**. 2013 Euromicro Conference on Digital System Design, Los Alamitos, CA, 2013, pp. 483-486.
- Noemie Beringuier-Boher, Marc Lacruche, David El-Baze, Jean-Max Dutertre, Jean-Baptiste Rigaud, Philippe Maurine. **Body Biasing Injection Attacks in Practice**. In Proceedings of the Third Workshop on Cryptography and Security in Computing Systems (CS2 '16), 2016.

Body Biasing Injection - Theory



Example - Previous Setups

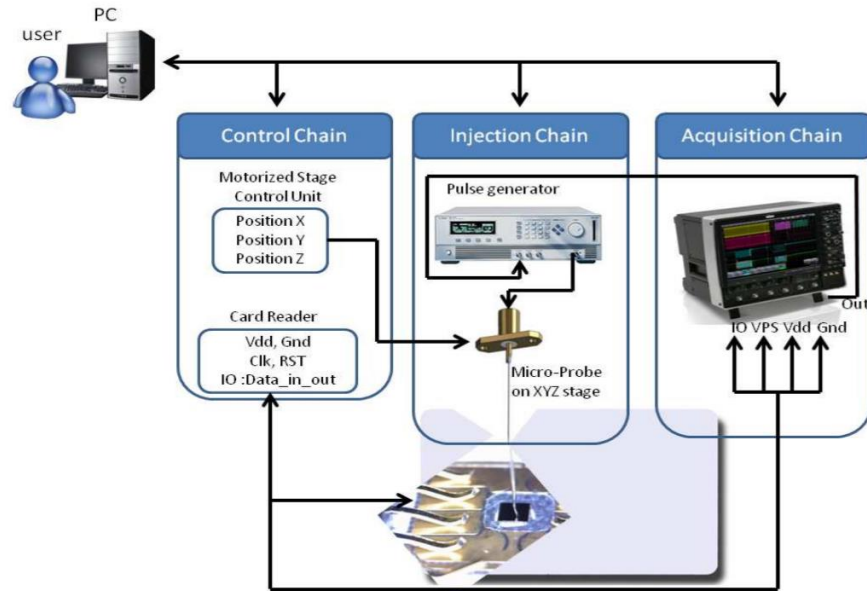


Figure4. Forward Body Bias Injection Platform

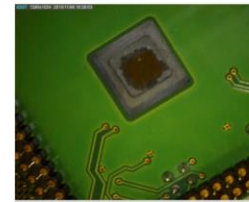


Figure 1: A backside opened micro-controller



Figure 2: A custom micro-probe with a $0.25\ \mu\text{m}$ diameter Tungsten needle

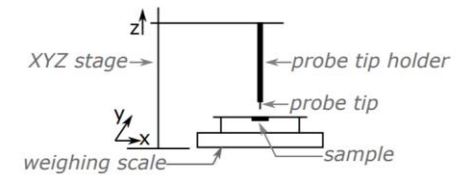
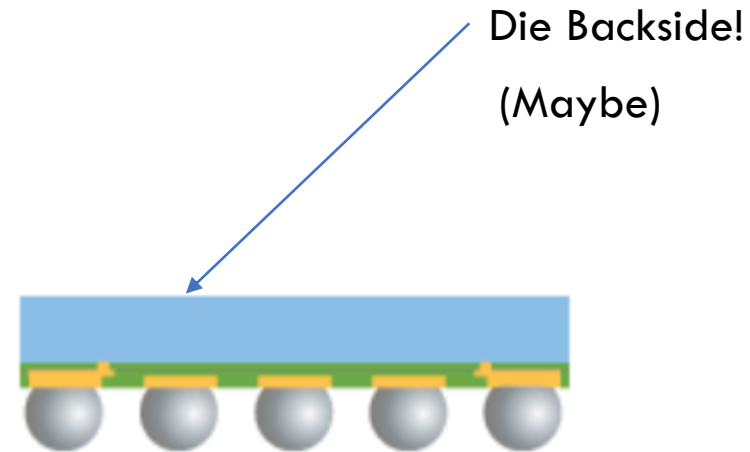
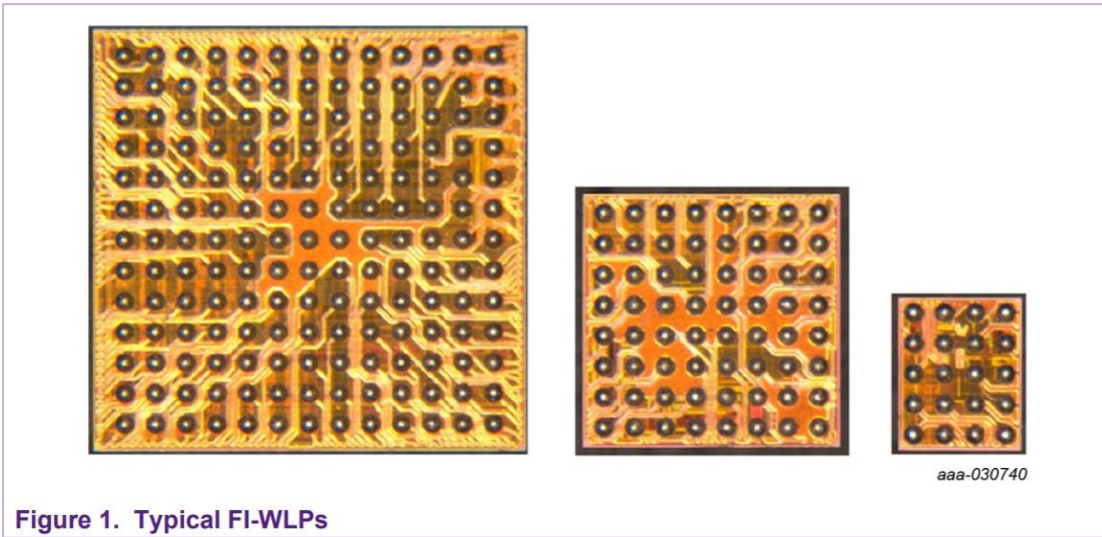


Figure 3: Schematic view of the attacked sample, the probe tip and the XYZ stage

tivity maps. Finally, to have a fully automated bench, a controlled Z axis is used to raise the tip and put it back after the XY move. The main difficulty in this task is to have a good Z axis repeatability. Indeed, the probe tip is made from Tungsten which is a hard material. Thus, the probe tip can easily damage the Device Under Test (DUT) if it is lowered back too far after being lifted and moved. Also, if the probe diameter is very small ($<100\ \mu\text{m}$ for example), the probe tip itself can be bent or broken. In order to address this, a weighing scale is added under the DUT to measure the pressure applied by the probe on the substrate. A weight variation of 80 g approximately achieves satisfying electrical contact between the probe and DUT (increasing the weight tends to deform the tip of the probe, thus increasing contact

Noemie Beringuier-Boher, Marc Lacruche, David El-Baze, Jean-Max Dutertre, Jean-Baptiste Rigaud, and Philippe Maurine. 2016. Body Biasing Injection Attacks in Practice.

What is Wafer Level Chip Scale Packaging (WLCSP)?



About that “Maybe”

Xenon Death Flash: a free physics lesson



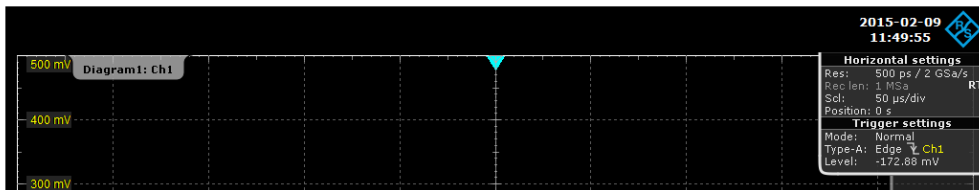
9th Feb 2015 Liz Upton 194 comments

If you own a Raspberry Pi 2, congratulations: you're also the proud owner of an elegant demonstration of the photoelectric effect!

At the weekend, Peter Onion, a veteran of our forums and of Raspberry Jams in Cambridge, Bletchley and surrounding areas (visible, costumed, in the background of [this photo](#) at the Christmas CamJam), discovered what we think might be the most adorable bug we've ever come across.

The Raspberry Pi 2 is camera-shy.

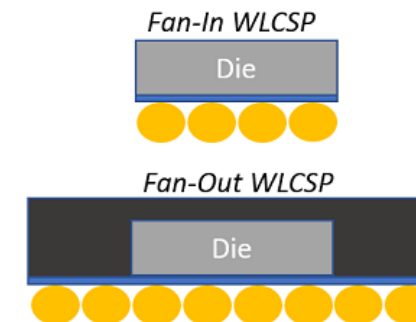
Peter's bug report came via our forums. He'd been proudly photographing his new Raspberry Pi 2, and had [discovered something peculiar](#): every time the flash on his camera went off, his Pi powered down.



<https://www.raspberrypi.org/blog/xenon-death-flash-a-free-physics-lesson/>

CARDIS 2020

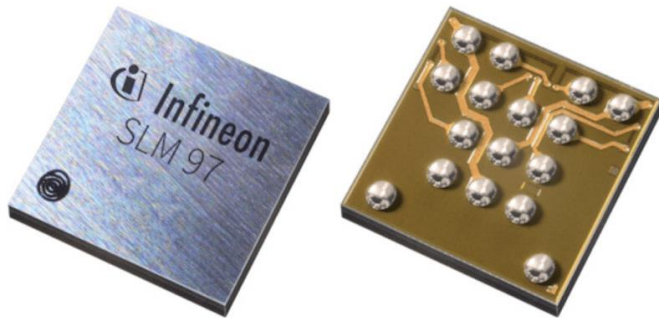
- For “non-security” reasons the backside may be covered somehow.
- On the STM32F415OG part, it's some covering (like paint?) that you can scrape off, or dissolve with acetone.
- On other devices they may have **small amount** of encapsulation around it for various reasons (including “fan-out” WLCSP which is closer to BGA):



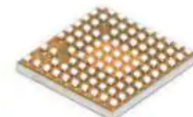

Where is WLCSP used in real life?

- Anywhere you need small devices!

Normal micros & secure micros available in this package.





nRF52840-CKAA-R



[Enlarge](#)

Images are for reference only
See Product Specifications

[G](#) [f](#) [t](#) [+](#)

Mouser #:	949-NRF52840-CKAA-R
Mfr. #:	nRF52840-CKAA-R
Mfr.:	Nordic Semiconductor
Customer #:	<input type="text" value="Customer #"/>
Description:	RF System on a Chip - SoC NRF52840 WLCSP
Lifecycle:	 New Product: New from this manufacturer.
Datasheet:	nRF52840-CKAA-R Datasheet (PDF)
ECAD Model:	 Build or Request PCB Footprint or Symbol

Download the free [Library Loader](#) to convert this file for your ECAD

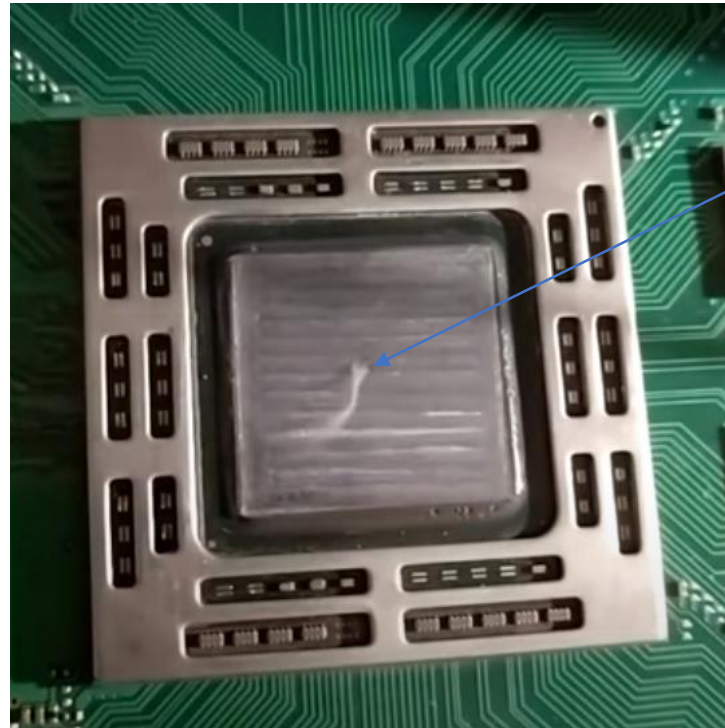
Why BBI?

From previous work, we know some nice things about BBI:

- X-Y positioning on chip surface means can fine-tune effect/location.
- Detectors for light will (hopefully) not be tripped by BBI.
- Detectors for EM may not be tripped by BBI (theorized, not clear if proven).

BBI Risks

- We also know - you can damage the IC.



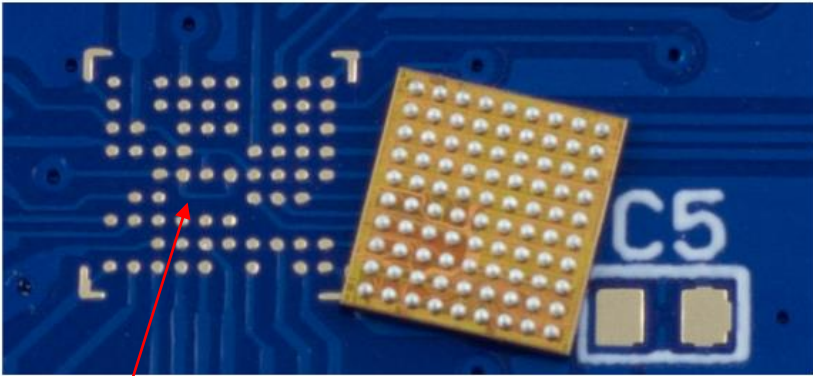
Magic Smoke!

Proof: Volodymyr Pikhur blowing up a PS4 in 2019: <https://www.youtube.com/watch?v=RK73RsU9a8A>

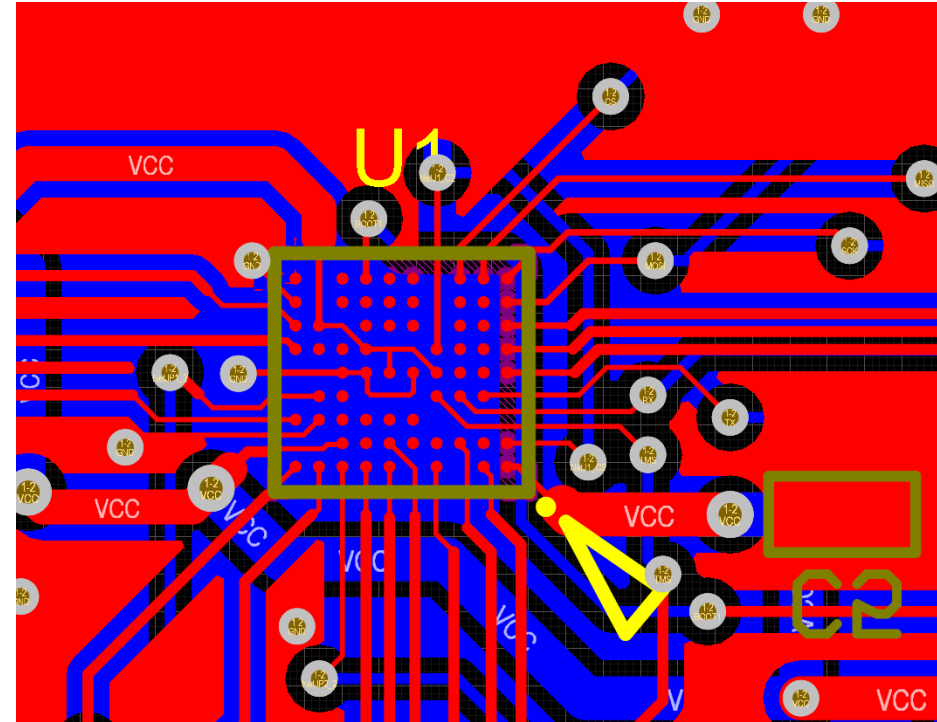
What makes BBI great?

- Building injection tools for BBI is much cheaper than for EMFI & laser.
- Using WLCSP (and other exposed die backside packaging) mean no chemicals.
 - **No need for hydrofluoric acid or similar!**
 - **Your skin/bones/eyes thank you!**

One Weird Trick to make WLCSP Work on standard 2-layer PCBs (engineers hate him!)

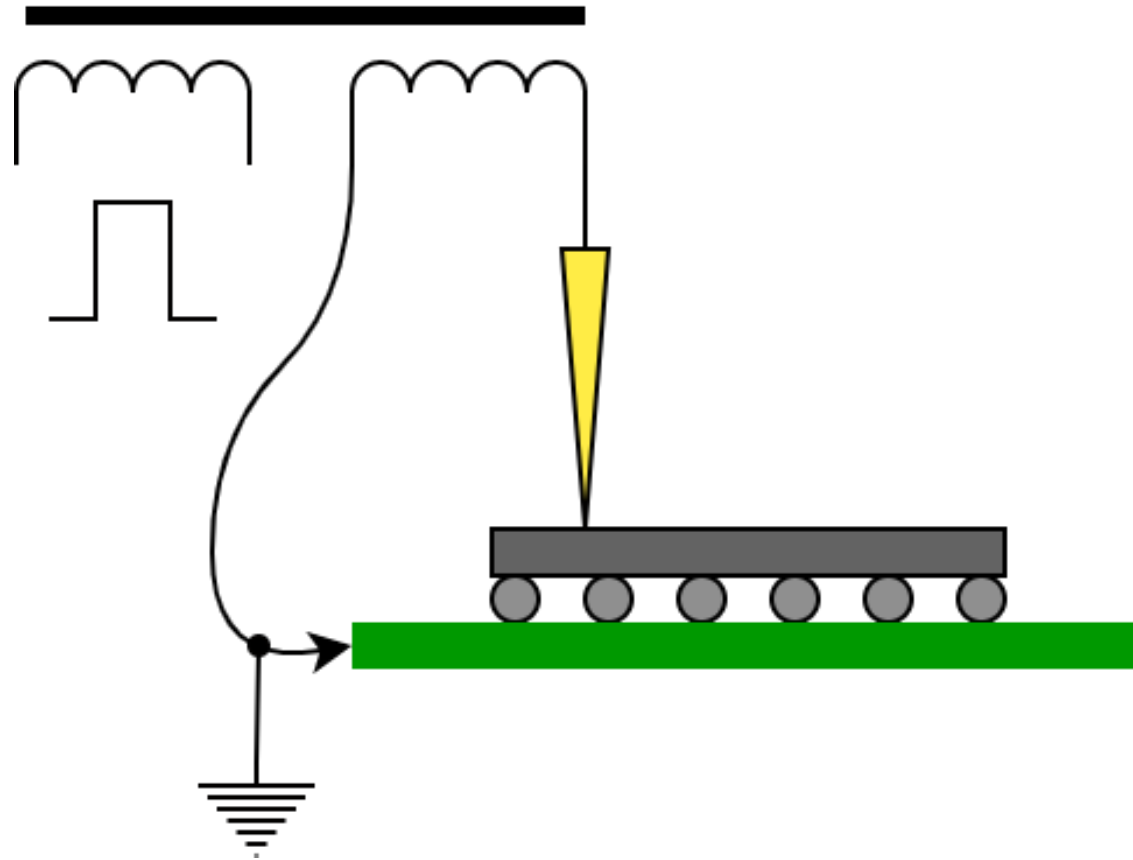


- Balls are removed to let traces route through – means there is no need for filled vias or microvias.
- Using 4 mil trace/space boards (now relatively low-cost)

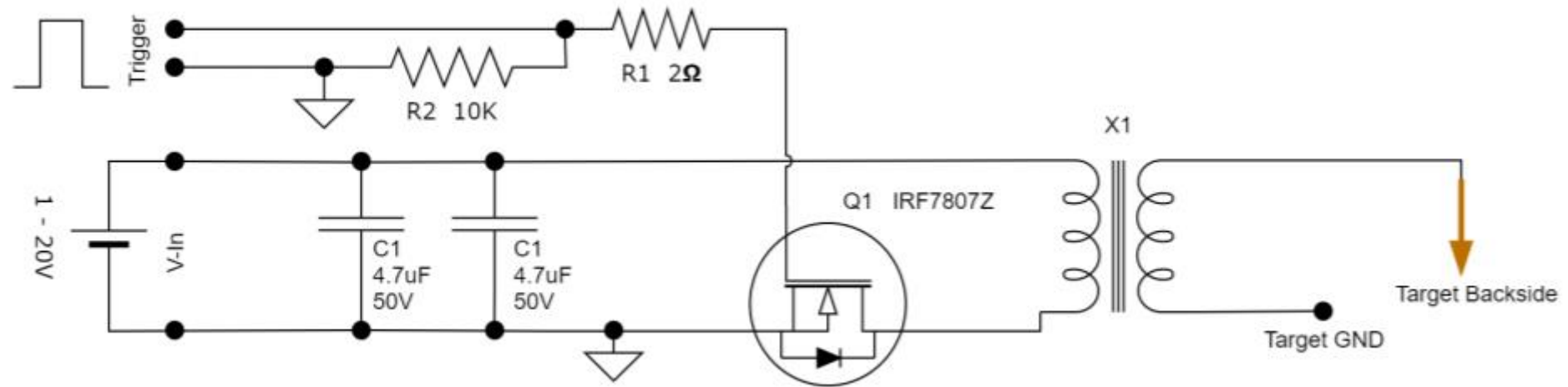


https://github.com/newaetech/chipwhisperer-target-cw308t/tree/master/CW308T_STM32F4_CSP

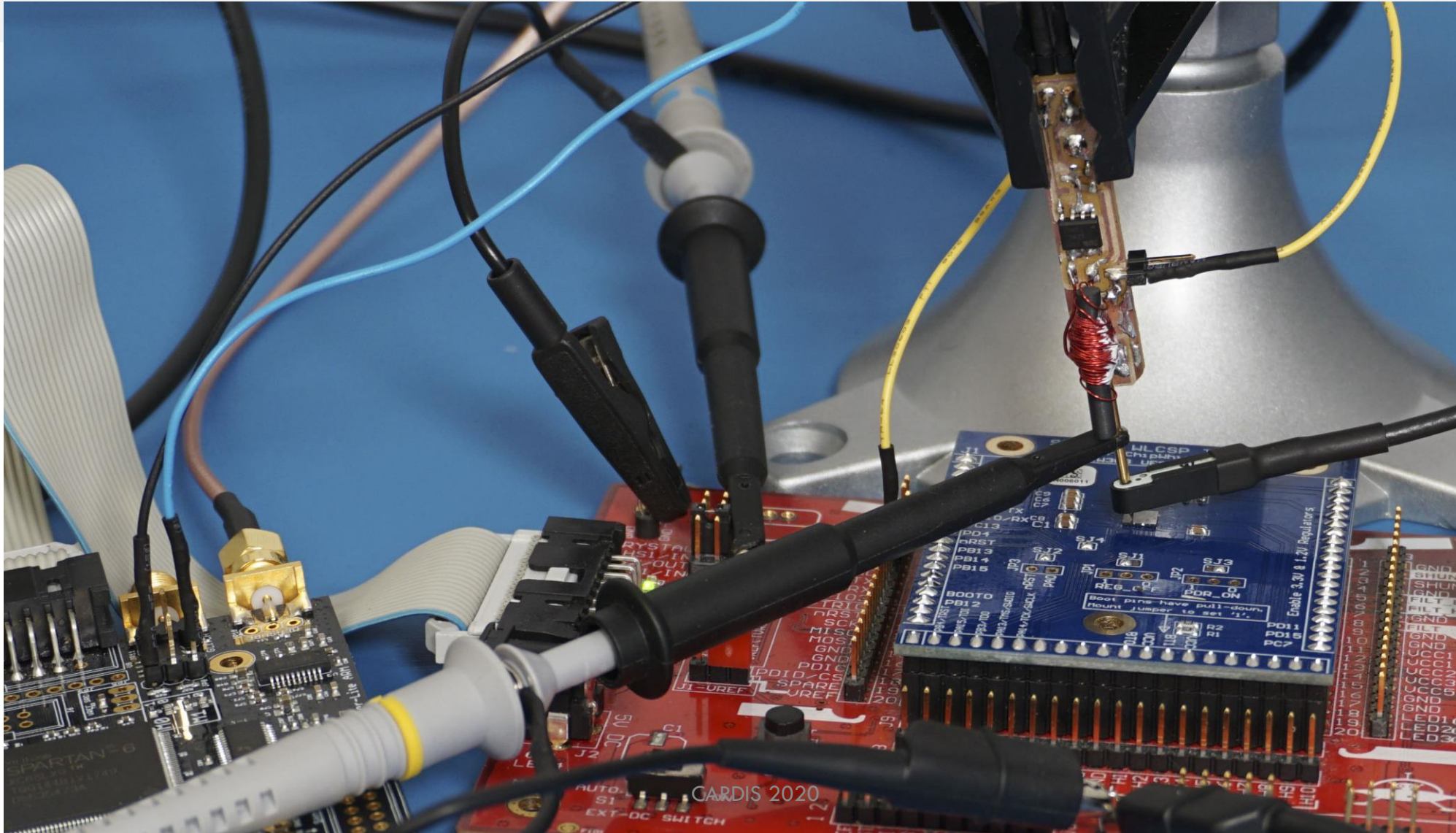
BBI using Transformer



Low Cost BBI Probe (that's it!)



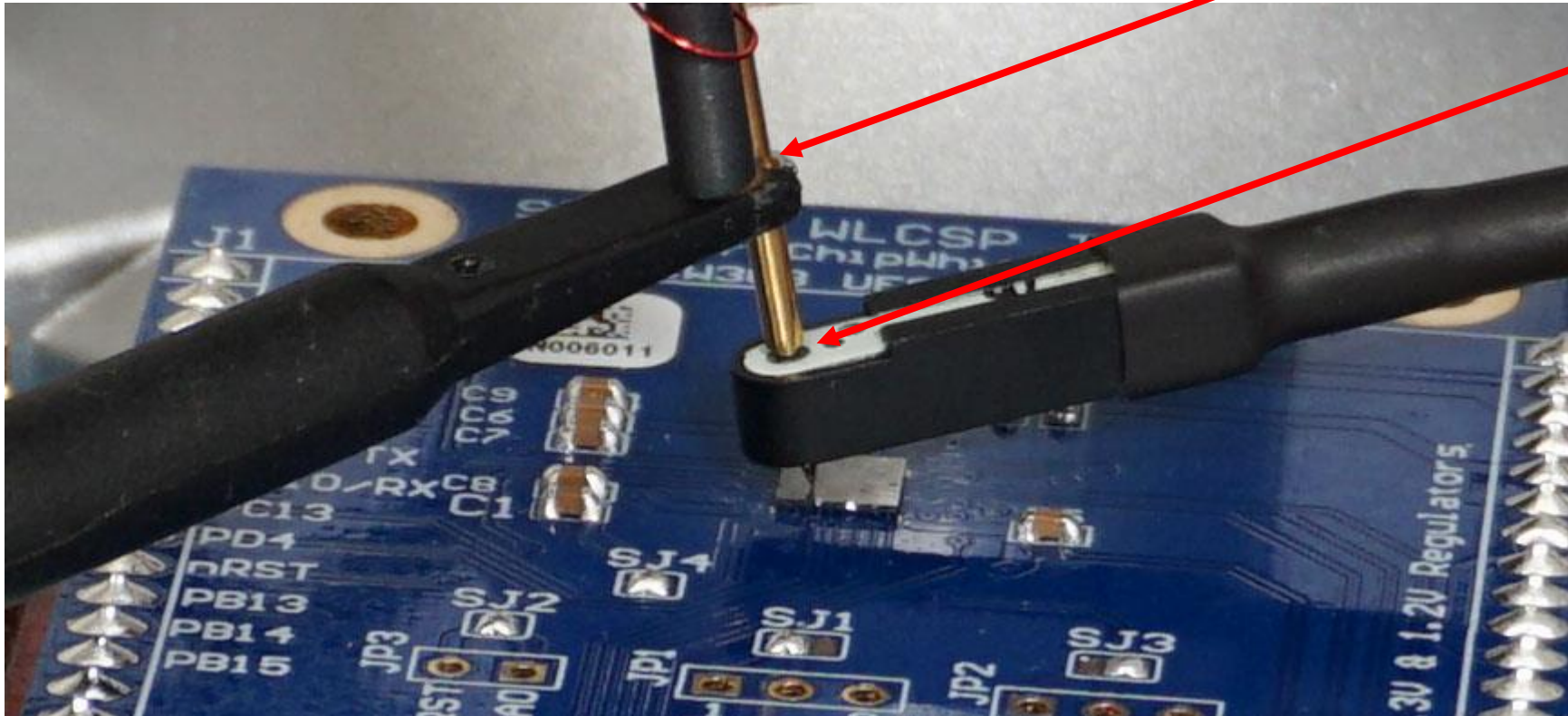
Implementation



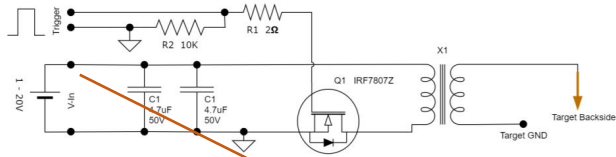
How to Characterize?

100:1 Passive scope probe.

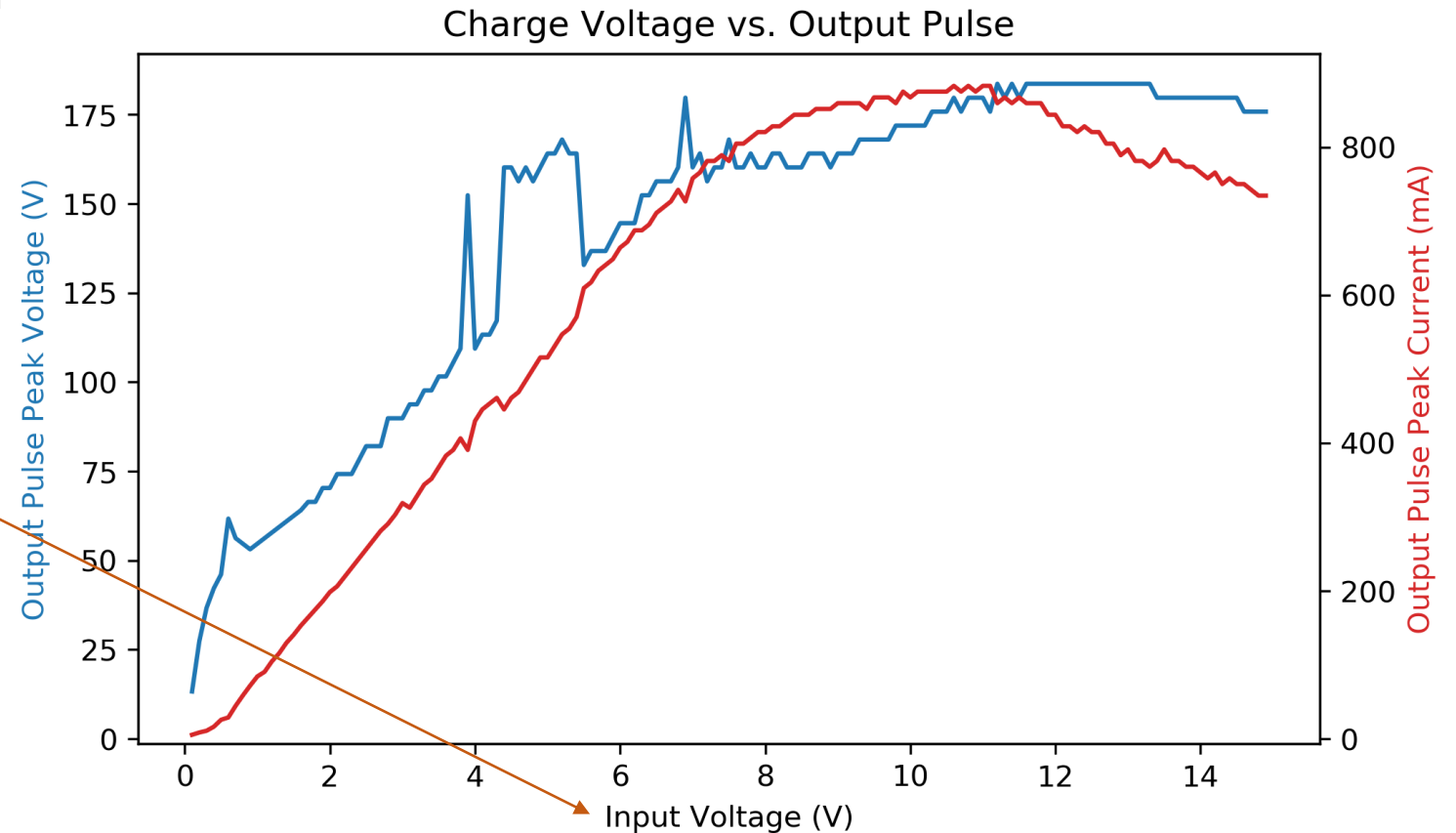
Tektronix CT-6 current probe (1 GHz BW).



Input Voltage → Output Voltage → Current



- We control the input voltage, and derive mapping to output voltage from measurements.
- Output voltage will change slightly with changes to impedance of target backside.
- All results in this paper will be for a given input voltage to the probe.



Input voltage is input to the BBI probe itself.

Die Backside Resistance?

- Using DMM, resistance ~ 200 kOhm (\rightarrow DC resistance).
- From voltage/current, resistance ~ 250 Ohm (\rightarrow AC impedance).
- Probably also related to break-down voltages etc, not purely a AC impedance function.
- **NOTE – Devices which blow up often showed very small resistance, suggesting that breakdown due to BBI is cause of failures...**

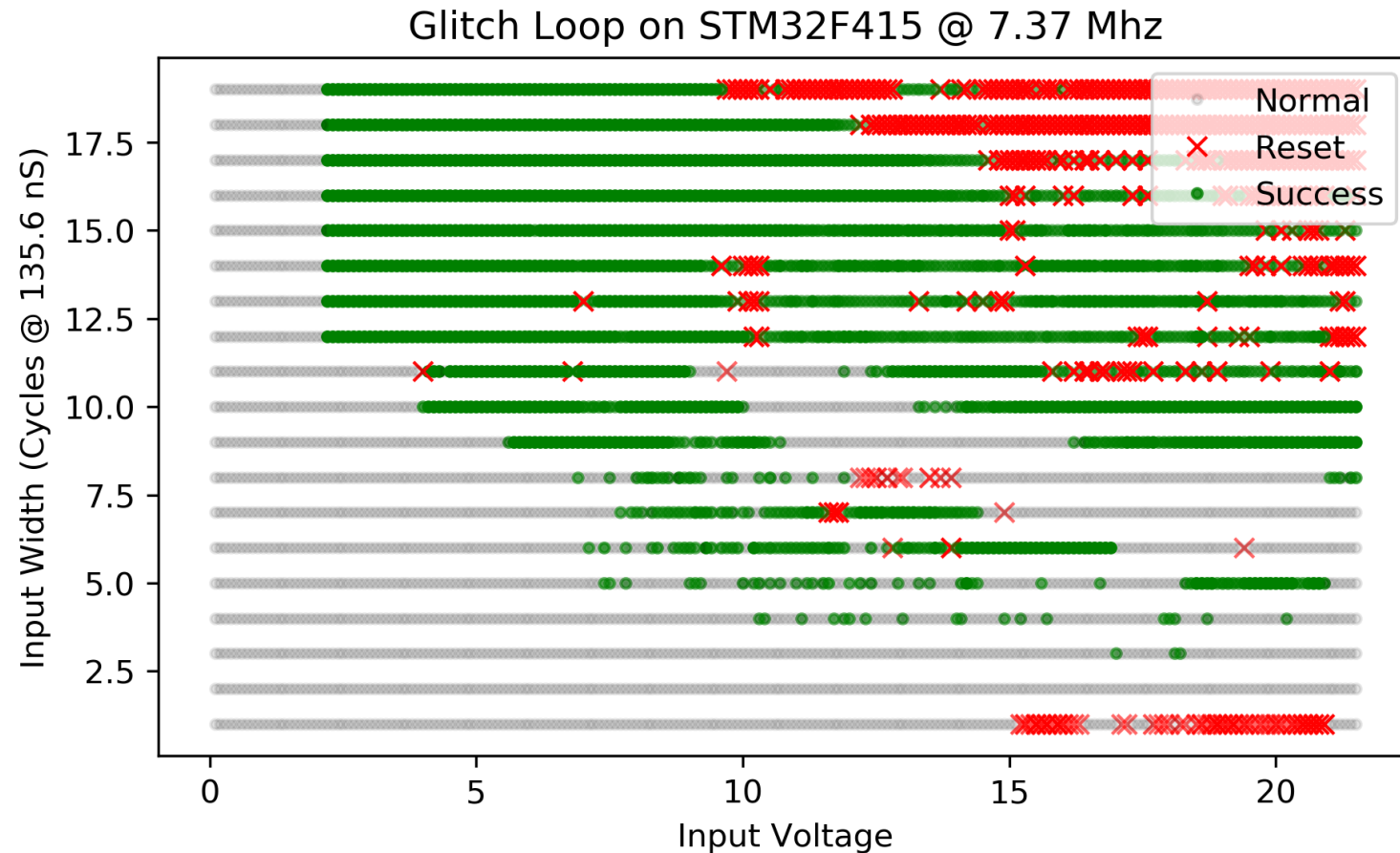
Glitch #1 – The Loop.

```
uint8_t glitch_loop(uint8_t* in)
{
    volatile uint16_t i, j;
    volatile uint32_t cnt;
    cnt = 0;
    trigger_high();
    for(i=0; i<50; i++){
        for(j=0; j<50; j++){
            cnt++;
        }
    }
    trigger_low();
    simpleserial_put("r", 4, (uint8_t*)&cnt);
    return (cnt != 2500);
}
```

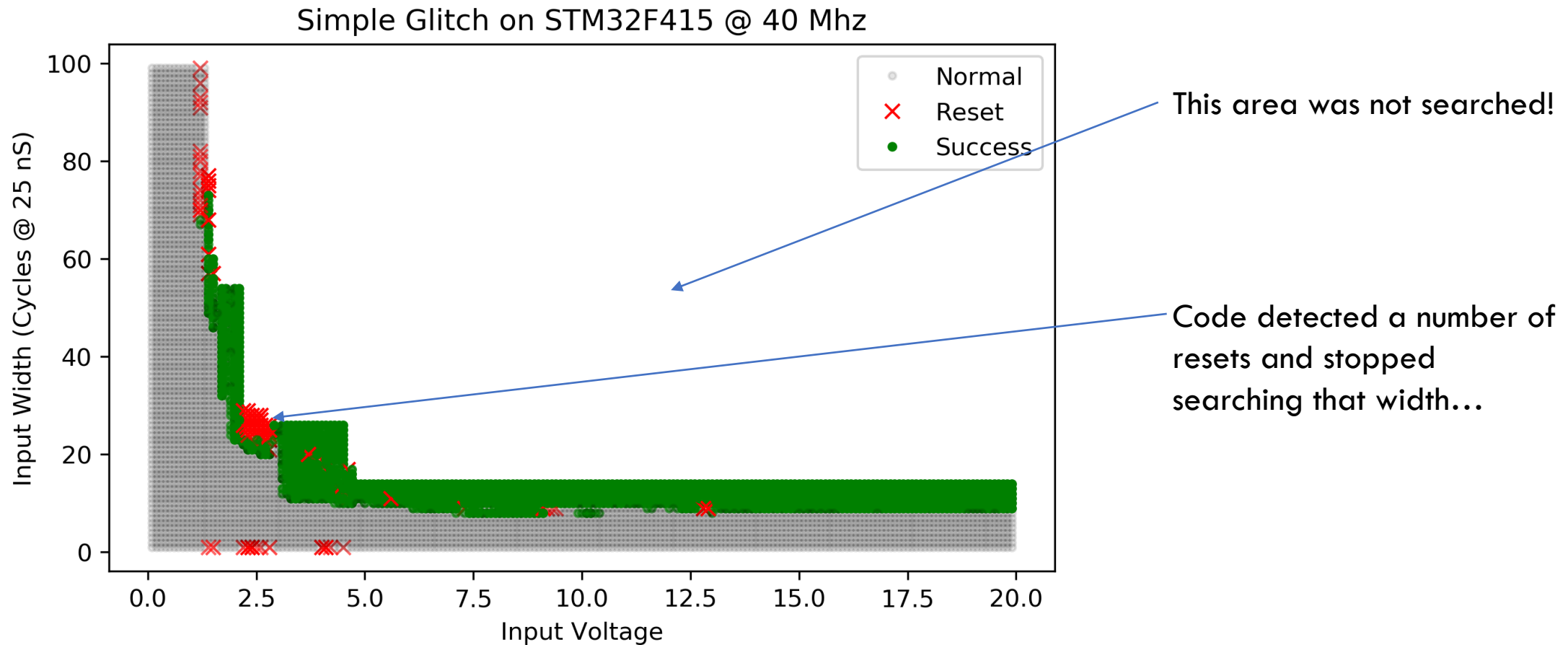
Easy trigger!

Lots of volatile variable accesses
etc.

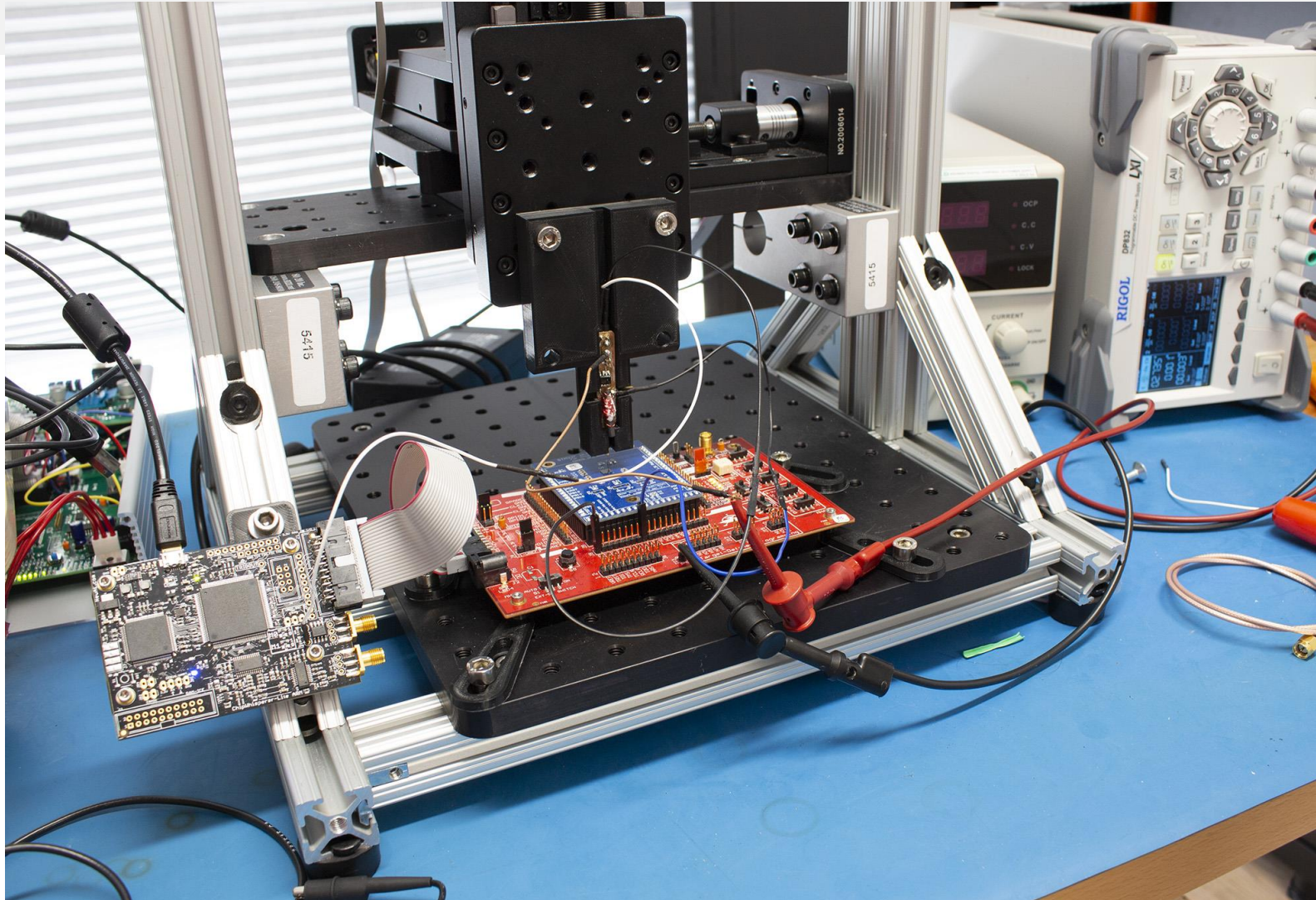
Running @ 7.37 MHz, Location 1



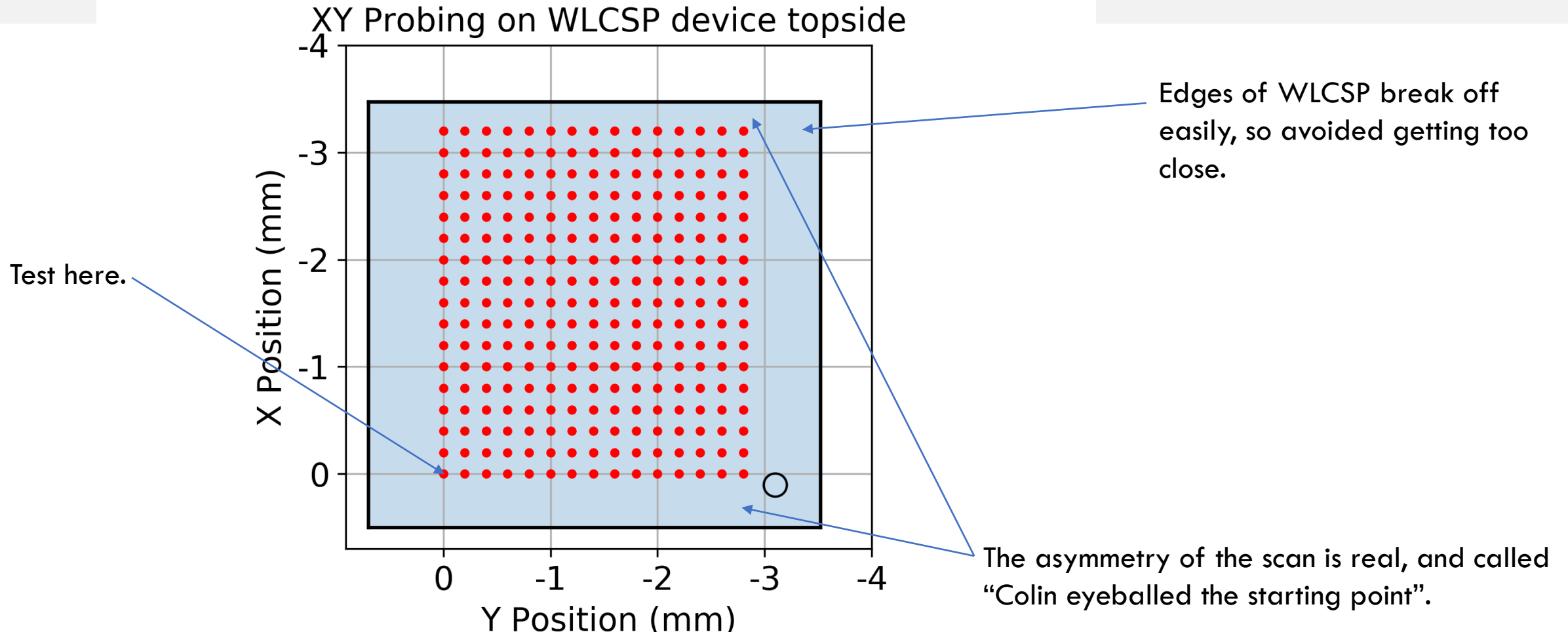
Running @ 40 MHz, Location 1



XY Positioning & what happens...



Scan Location



One Red Dot = Lots of Scanning

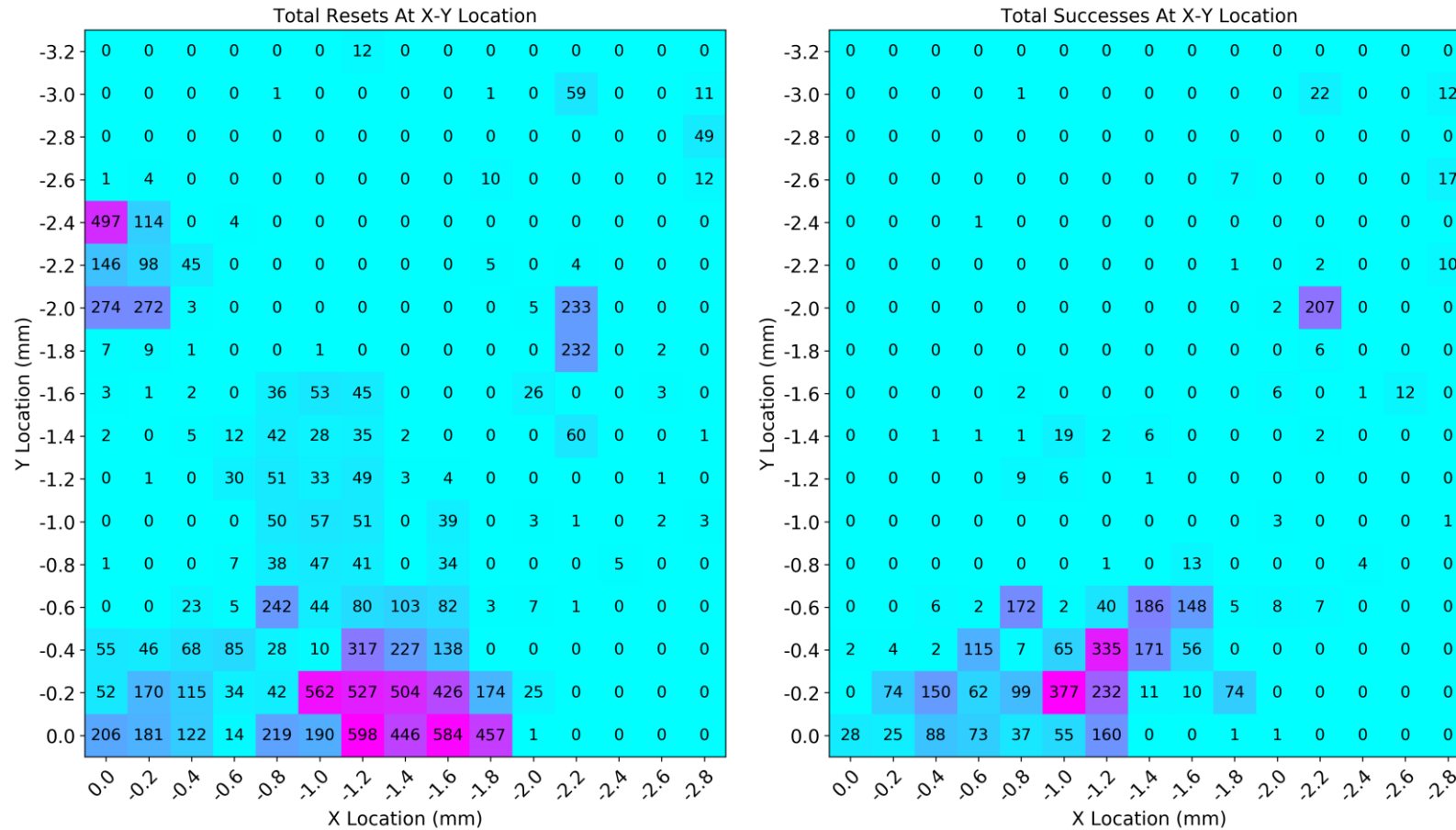
Algorithm 1: Summary of evaluation at each step.

Result: Fault injection details about location.

```
for TrialNumber ← 1 to 2 do
  for BBIV ← 0.50V to 10.00V do
    for GlitchCycles ← 5 to 60 do
      Run double-loop code;
      Insert pulse;
      if Output is incorrect but correctly formed then
        Log success;
      end
      if Reset is detected or output malformed then
        Log reset;
      end
      if 5 or more resets in a row then
        Check for flash corruption;
        if Flash memory corruption then
          Log flash error;
          Reprogram target;
        end
      end
    end
  end
end
end
```

2090 combinations.

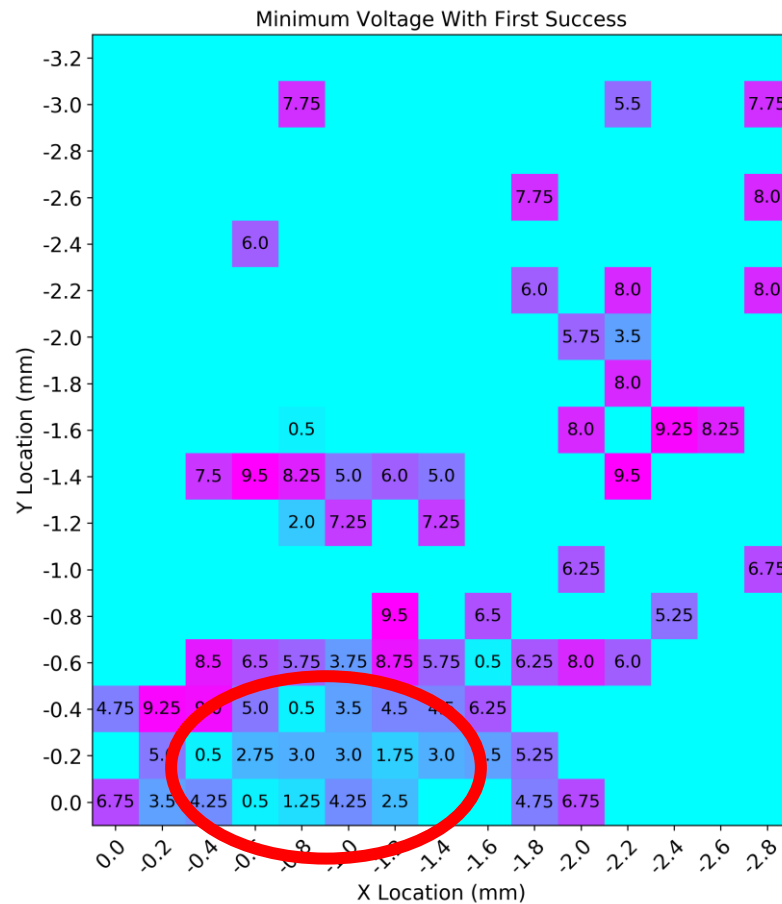
Example: Reset v. Successes



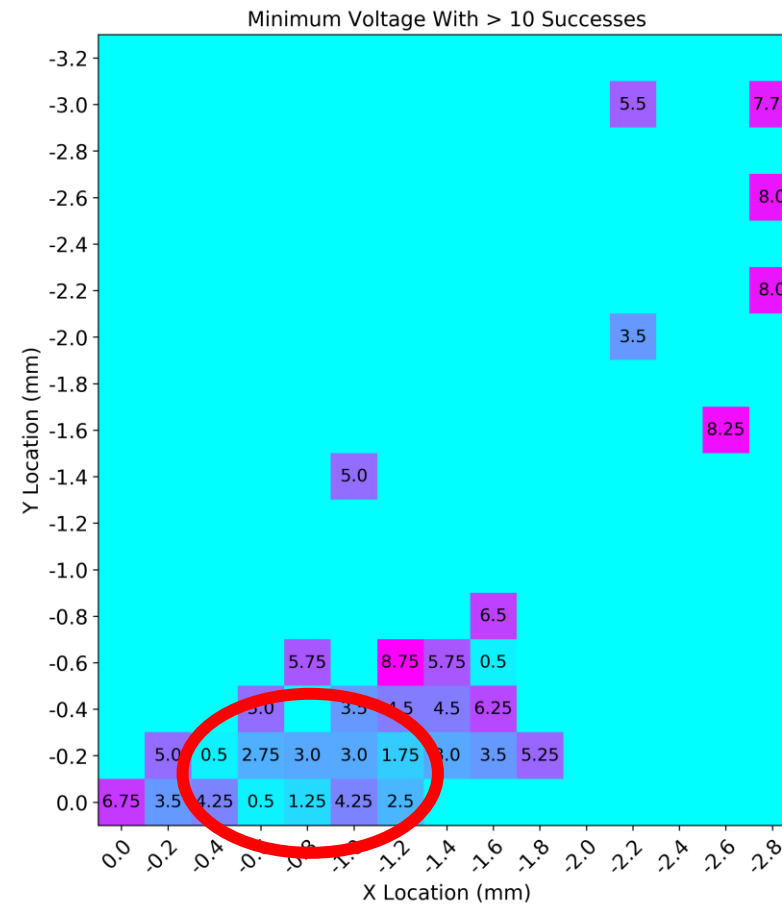
Counts totals across all settings of voltage.

Example: Minimum input voltage for...

When *first* glitch is seen:



When a *reliable* glitch is seen:



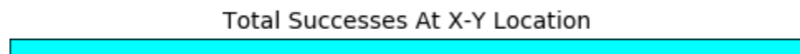
What else can you do?

- The full dataset & notebooks to create those graphs have been pushed to github, a lot more data you can plot!

```
plt.setp(ax.get_xticklabels(), rotation=45, ha="right",
         rotation_mode="anchor")

# Loop over data dimensions and create text annotations.
for i in range(len(y)):
    for j in range(len(x)):
        text = ax.text(j, i, hmdata[i, j],
                       ha="center", va="center", color="k")
ax.set_xlim(ax.get_xlim()[::-1]) # Flip x
return ax

ax = makehmap(successes)
ax.set_title("Total Successes At X-Y Location")
plt.savefig("successesxy.png", dpi=300, bbox_inches = 'tight')
plt.show()
ax = makehmap(resets)
ax.set_title("Total Resets At X-Y Location")
plt.savefig("resetsxy.png", dpi=300, bbox_inches = 'tight')
plt.show()
```



<https://github.com/newaetech/chipjabber-basicbbi/tree/master/cardis2020>

Glitch #2 - RSA

- Original BBI paper used RSA glitch to understand sensitivity... recreated here.
- MBED-TLS RSA implementation, which uses CRT.

```
1251
1252     /* Compare in constant time just in case */
1253     for( diff = 0, i = 0; i < ctx->len; i++ )
1254         diff |= verif[i] ^ sig[i];
1255     diff_no_optimize = diff;
1256
1257     if( diff_no_optimize != 0 )
1258     {
1259         ret = MBEDTLS_ERR_RSA_PRIVATE_FAILED;
1260         goto cleanup;
1261     }
```

RSA implementation checks returned signature is valid.

...we removed that check.

- **Claimed** – the original BBI paper used a single fault CRT as well.
- **In reality** – because I'm lazy.

Glitch #2 - Results

RSA Signing Result	Occurrence
Exploitable (p or q)	54.2 %
Device Reset	28.0 %
Normal	11.9 %
Invalid	5.9 %

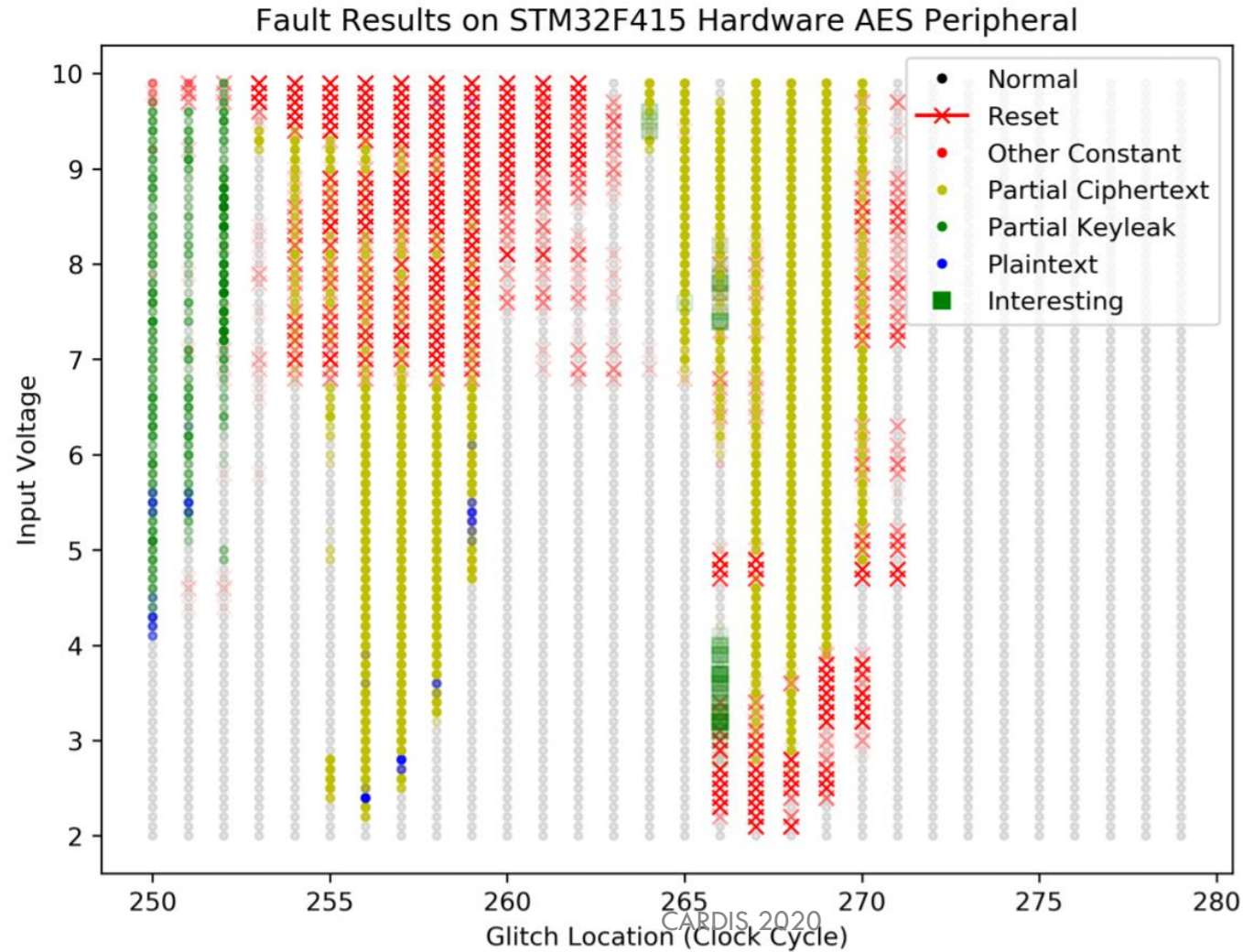
Table 1: RSA-CRT Fault Attack.

Glitch #3 – DFA on Hardware AES Engine

Response types:

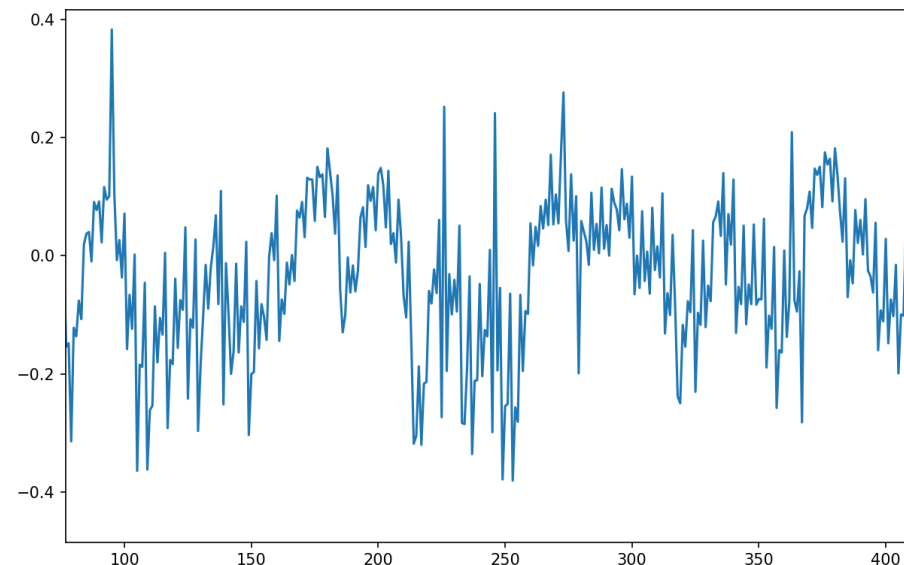
- Key leakage
 - Obvious portions of key copied to returned value.
- Partial ciphertext
 - Ciphertext output, but with constants in some bytes/words (normally 0).
- Plaintext leakage
 - Portions of plaintext returned instead of ciphertext.
- Other constant output
 - No obvious dependance on inputs (i.e., all 0s).
- Interesting-looking output
 - Some incorrect output not the above cases. Typically looks like random output.

Glitch #3 – Results



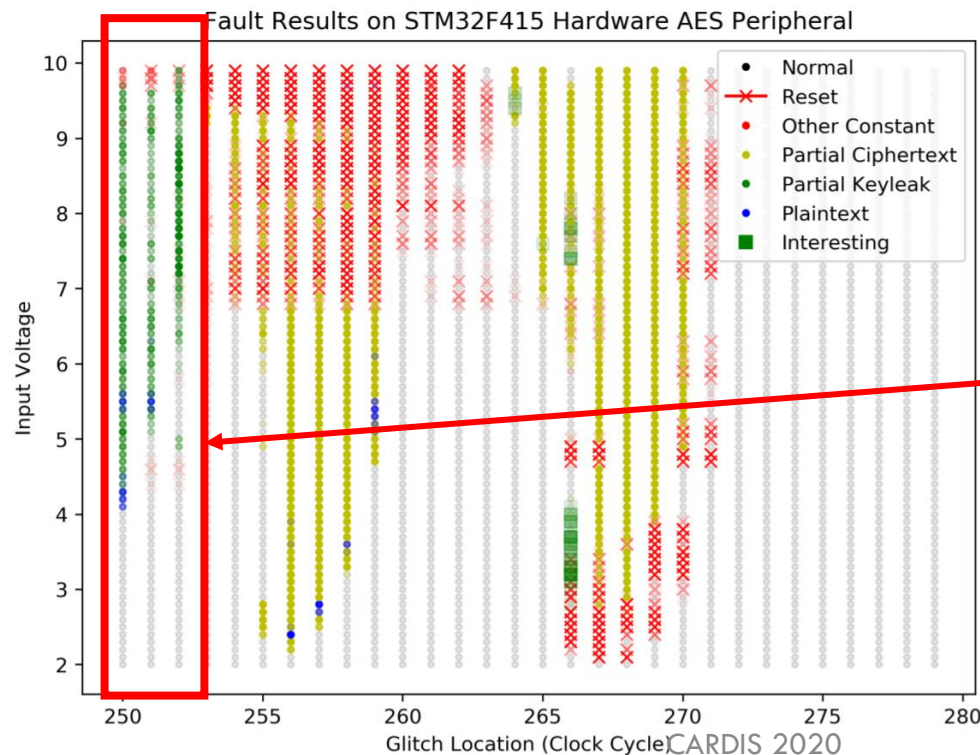
Hardware AES Timing

- STM32F4 HW AES leakage is known (classic “last-round HD of state”).
- We can perform CPA attack to confirm when actual engine is running.
- Engine appears to be running with 1 cycle per round.



Reasons for Key & Plaintext Leakage

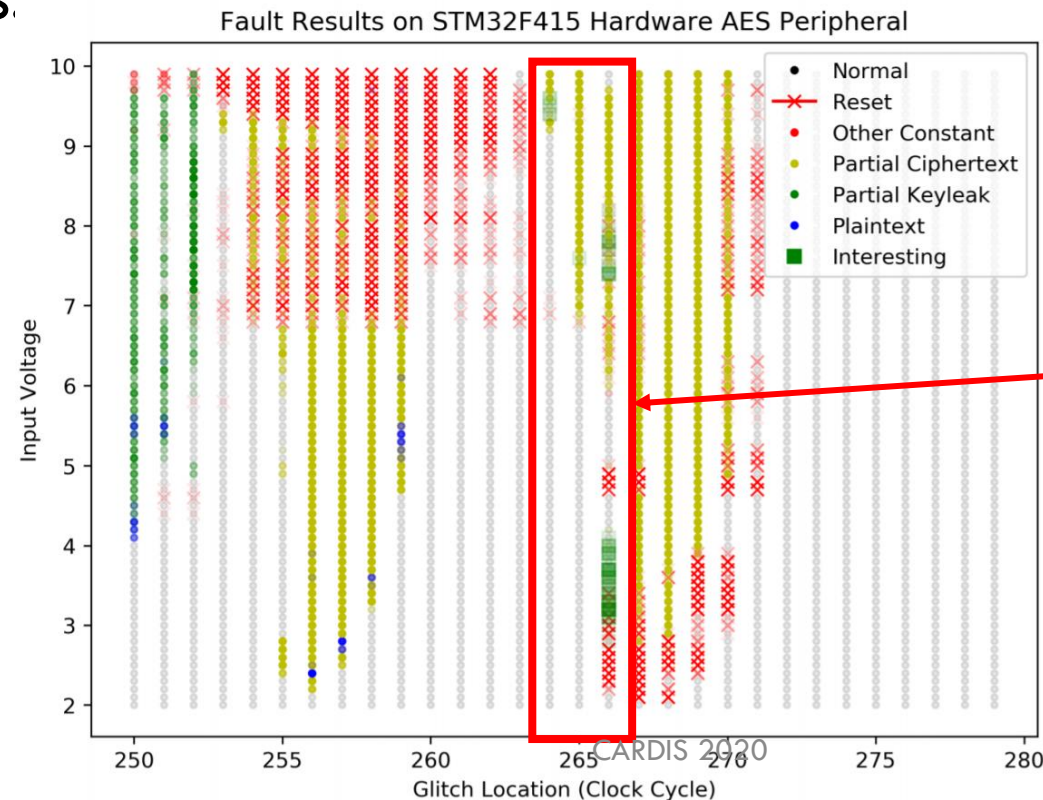
- At some point – data is transferred to/from AES core.
- Assumed this leakage is actually failure to unload correctly.



Location of “key leakage” is early on – maybe entire encryption skipped?

What is interesting?

- Did not match known (obvious) fault models.
- Reversing output and comparing intermediate rounds did not reveal any matches.



This time is during the actual encryption – suggests more than just input/output corruption?

Other Result – Flash set to 0?

```
n [311]: #Test board #1
```

```
# Erase - then re-run super-Loop
data3 = stm32f.cmdReadMemory(0x80000000, 0x100)
print(data3)
```

[illegible]

7 Differing amounts of bytes
set to "0"!

```
[n [295]: #Test board #1
```

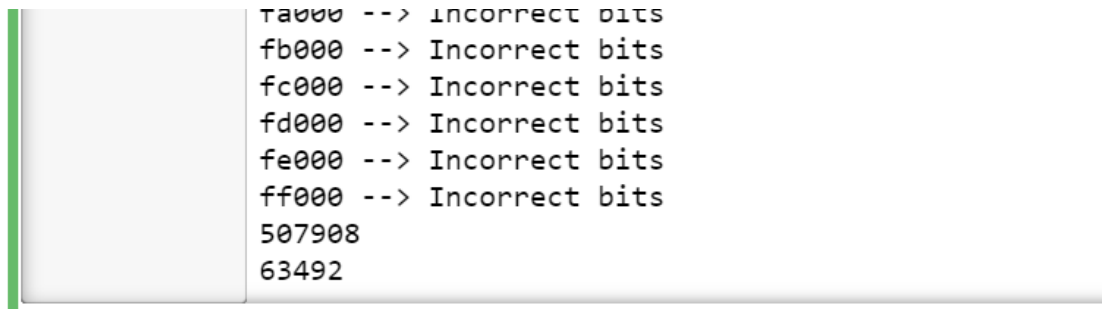
```
#Testing after a serious FI campaign...
#data1 = stm32f.cmdReadMemory(0x8000000, 0x100)
#data2 = stm32f.cmdReadMemory(0x8000100, 0x100)
print(data1)
print(data2)
```

[illegible]

Assume we trip some
“programming” routine?

Other Result – Flash Damage

- One test device would only partially erase on half the flash.
 - Performing a chip erase would mean a small percentage of bits read as 0 (not 1 as intended).
 - The bits would random change when re-read (which read as 0).
 - Setting bits to 0 worked perfectly.
 - The other half of the flash memory worked perfectly.
 - No excessive power consumption (see next slide for other ways to damage).



```
ta000 --> Incorrect bits
fb000 --> Incorrect bits
fc000 --> Incorrect bits
fd000 --> Incorrect bits
fe000 --> Incorrect bits
ff000 --> Incorrect bits
507908
63492
```

- Assumed damage to charge pump or control circuitry for the flash?

Other Result – Device Destruction.

- Several devices got very hot, and mostly didn't work anymore.
 - Interestingly some did try to work even in this state.
 - Program/erase didn't work reliably with these “too hot” devices.
 - Power consumption was very high – may be just “browning out”.

Bringing Body-biasing Injection Back

- BBI is not new – but appears to have somewhat languished from lack of accessible tooling.
- **ChipJabber-BasicBBI** tries to fix that – very low cost & open-source.

<https://github.com/newaetech/chipjabber-basicbbi>

<https://discord.gg/chipwhisperer> Join my discord for hardware security chat!

[@colinoflynn](#) (twitter)

colin@oflynn.com (email)