Small Computers Are Bad And You Should Feel Bad

Artemis Tosini
artemist@cmu.edu

Über mich

- PPP member
- CMU Student
  - Freshman in Electrical and Computer Engineering
- Have been messing with electronics for far too long
- Artemis Tosini <artemist@cmu.edu> (she/her)
  - 3D2B B230 F9FA F0C5 1832 46DD 4FDC 96F1 61E7 BA8A

Overview

- Your goals when hacking hardware
- What hardware you're working with
- Hacking hardware
  - "designers not caring"
  - Studying what the hardware is doing
  - Power analysis
  - Fault injection

Why would you want to hack hardware?

- Hardware controls fun physical things
  - Like doors, cars, and nuclear missiles
- People connect things to the internet that they *really* shouldn’t
- People keep secrets in hardware
  - See: Every smart card ever

Things you might want to do

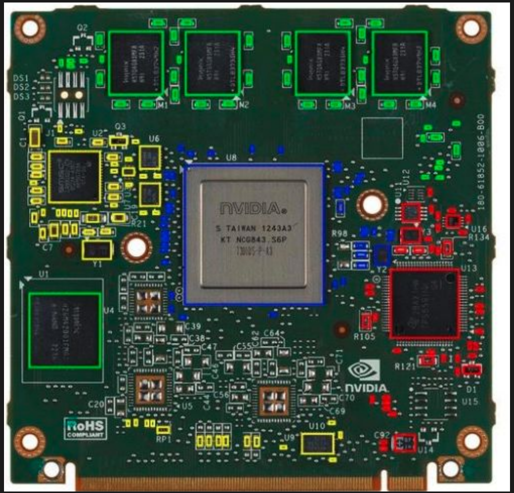
- Get a copy of code
- Get copies of private data (e.g. on smart cards)
- Get code execution on the device
- Gain persistence on the device

How do you hack?

- You can always just hack the the software
  - It’s basically the same as hacking other things, but it’s ARM or something
  - There’s lots of resources on this, you don’t need me
- Or you can abuse properties of the hardware
  - This route is frequently ignored when protecting things
  - It’s more interesting[citation needed]

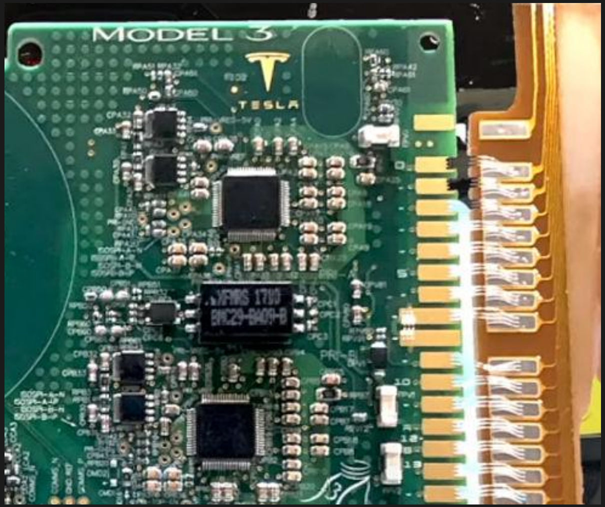
Small computers (System on Module)

- Pretty normal computers
  - Except using ARM/MIPS/something weird
- Might run Linux, Windows, etc.
- Game consoles, Phones, and IoT things
  - Less critical car systems also fit
- Uses a SoC (RAM, flash external)



Smaller computers (microcontrollers, µCs)

- Single-chip, has flash, RAM, CPU
- Very little computing power
- Generally for dedicated functions
- Bare-metal or running an RTOS
- Used everywhere
  - see BadUSB
- Arduino is based around one



Considerations with Microcontrollers

- Microcontrollers use different architectures
  - AVR or ARM Thumb
- Many microcontrollers have no MMU
- Some are physically unable to execute from RAM
  - These µCs are a Harvard architecture

When You Can’t Hack Software

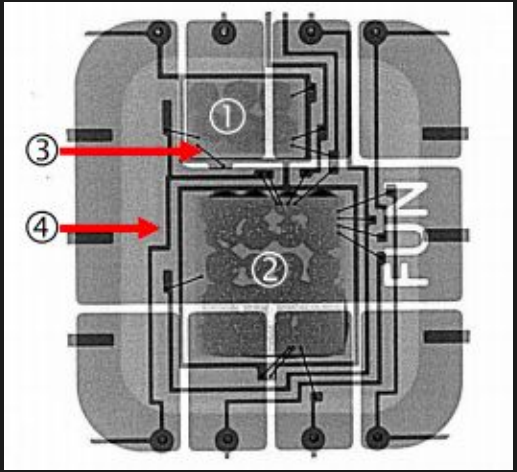
- People can leave open unintended debug ports
  - JTAG gives you complete access
  - UART just gives you a terminal
- Sometimes, you need something invasive
  - Intercepting busses
  - Side-channel attacks
  - Glitching

Intercepting busses

- Chips communicate using standard busses
- This could contain secret keys
- You can also modify commands on the wire
- Helps get code, private data, code execution, and other attacks

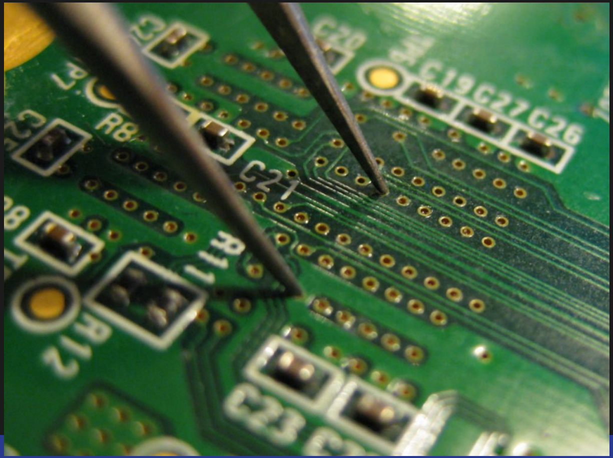
EMV interception attack

- EMV cards require a PIN
- This check was enforced by the terminal
- Make “PIN check” always return true
- Use a microcontroller to intercept



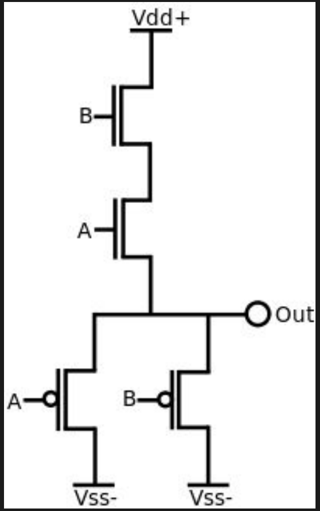
Tweezer Attack

- RAM used in the Wii has separate address pins
- You can access a limited part of memory
- You can bypass this by externally forcing pins to be true



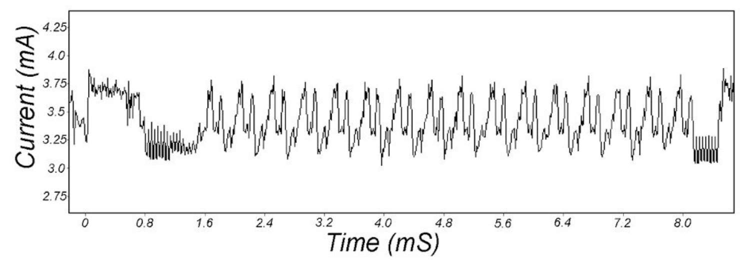
Sidechannel analysis

- A gate consumes power when it switches
  - This turns into EM radiation, heat, and sound
- Give you a picture of what a computer is doing
- This can tell you e.g. multiplication vs addition
- Can use this to extract private keys
  - Also for timing of fault injection



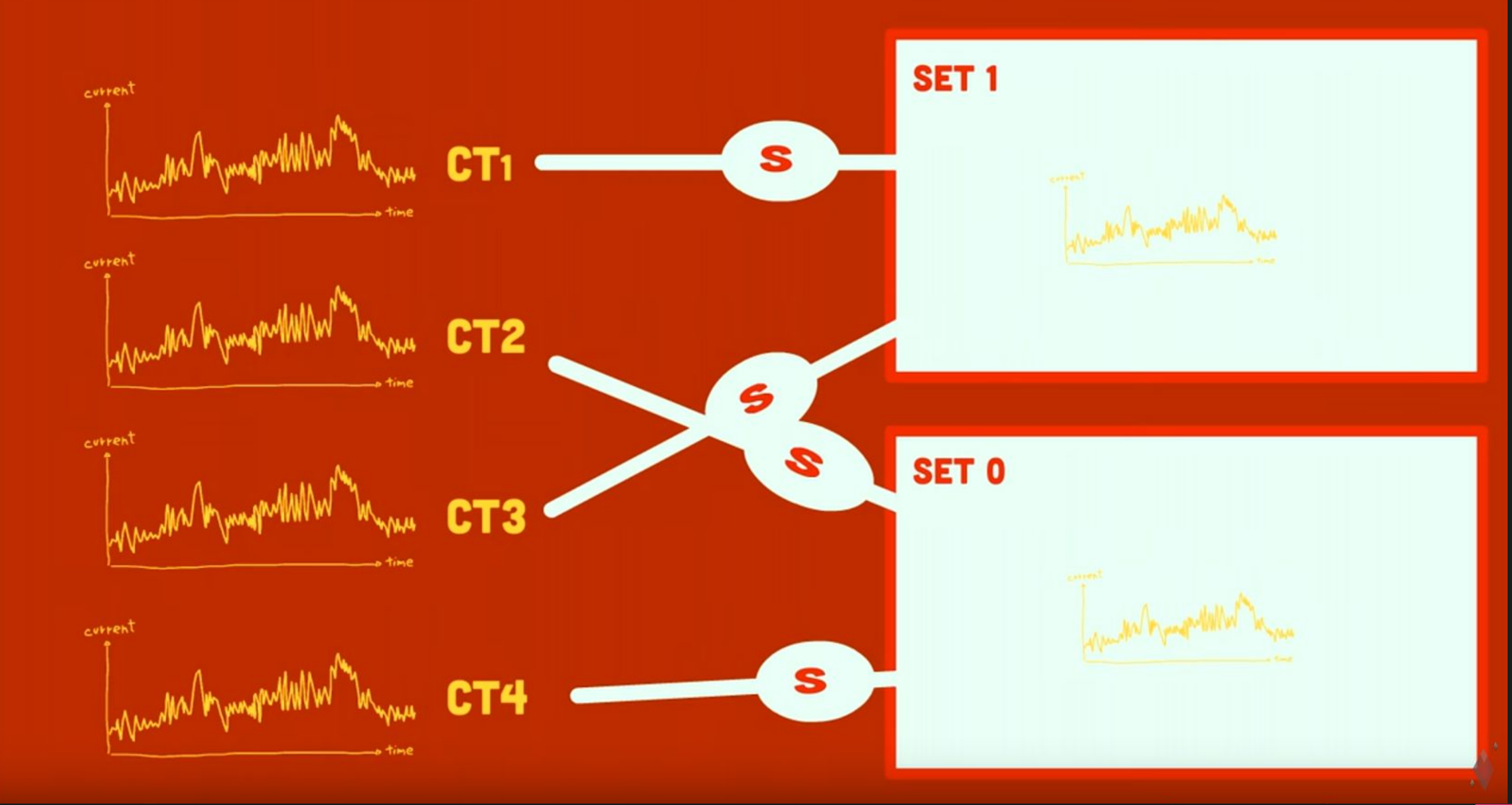
GPG key extraction

- Laptop PSUs emit coil whine
- GPG did not properly blind RSA
- Get bits of the key from each decryption
- With enough samples, get the private key



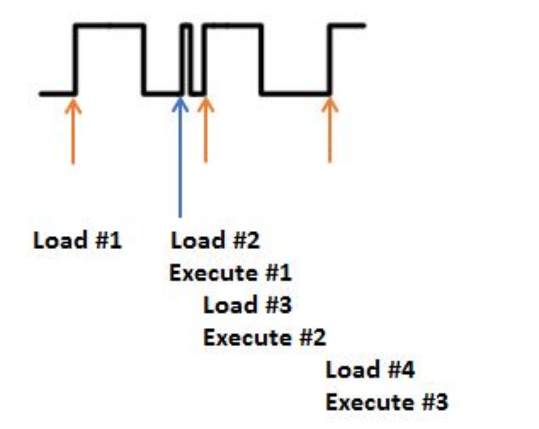
Smart Card

- Store secret RSA, ECC, AES, or DES keys
- Use Differential Power Analysis to get keys
  - Need many samples and ciphertexts
- guess key, combine traces, test hypothesis, repeat


current
time
CT1
CT2
CT3
CT4
S
S
S
S
SET 1
SET 0

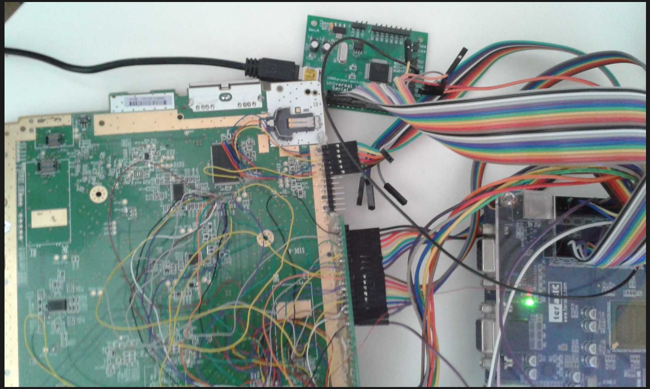
Fault Injection/Glitching

- Logic needs voltage and time
- Unexpected results without this
- Useful to change a single jump
  - For example, if(memcmp(...)), or if (len < …)
  - Can give everything but persistence
- A single fault can also reveal keys
  - Using some math I don't understand



Nintendo Wii U keys

- Boot0 is chain of trust, decrypts later stages
- Not buggy, locks keys
- Does a bounds check
- Glitch used for conditional branch
- Overwrote code, dumped keys



Defense

- It’s really hard
- Cleaning up low hanging fruit is easy
  - Don’t leave a JTAG port enabled on the chip
- Some (very expensive) µCs have 2 cores check each other
- Make algorithms harder to analyze