BornHack 2023 NFC Badges

Thomas Flummer, BornHack 2023

What is NFC

Near Field Communication

- Uses 13.56 MHz (also know as HF RFID)
- Short range (usually a few centimeters)
- Low datarate

Other RFID types

and some of the common uses of these

- < 135 kHz (LF)
  - Older access control (often 125 kHz)
  - Animal tracking
- 433 MHz/900 MHz (UHF)
  - Shipping container location
  - Manufacturing
- 2.4GHz
  - Toll booth systems

Where is NFC used

some of the more common ones

- Payment cards (contactless)
- Travel cards
- Access control
- Information (POI/museum/smart posters)
- Provisioning (consumer electronics configuration)
- Trigger for tasks on a device

How does it work

Active device

- Usually a phone, payment terminal, door reader, transit gate, etc.
- Has power (battery, plugged in, etc.)
- Creates a "powered field"

Passive tags

Cards, keychain bricks, stickers, etc.

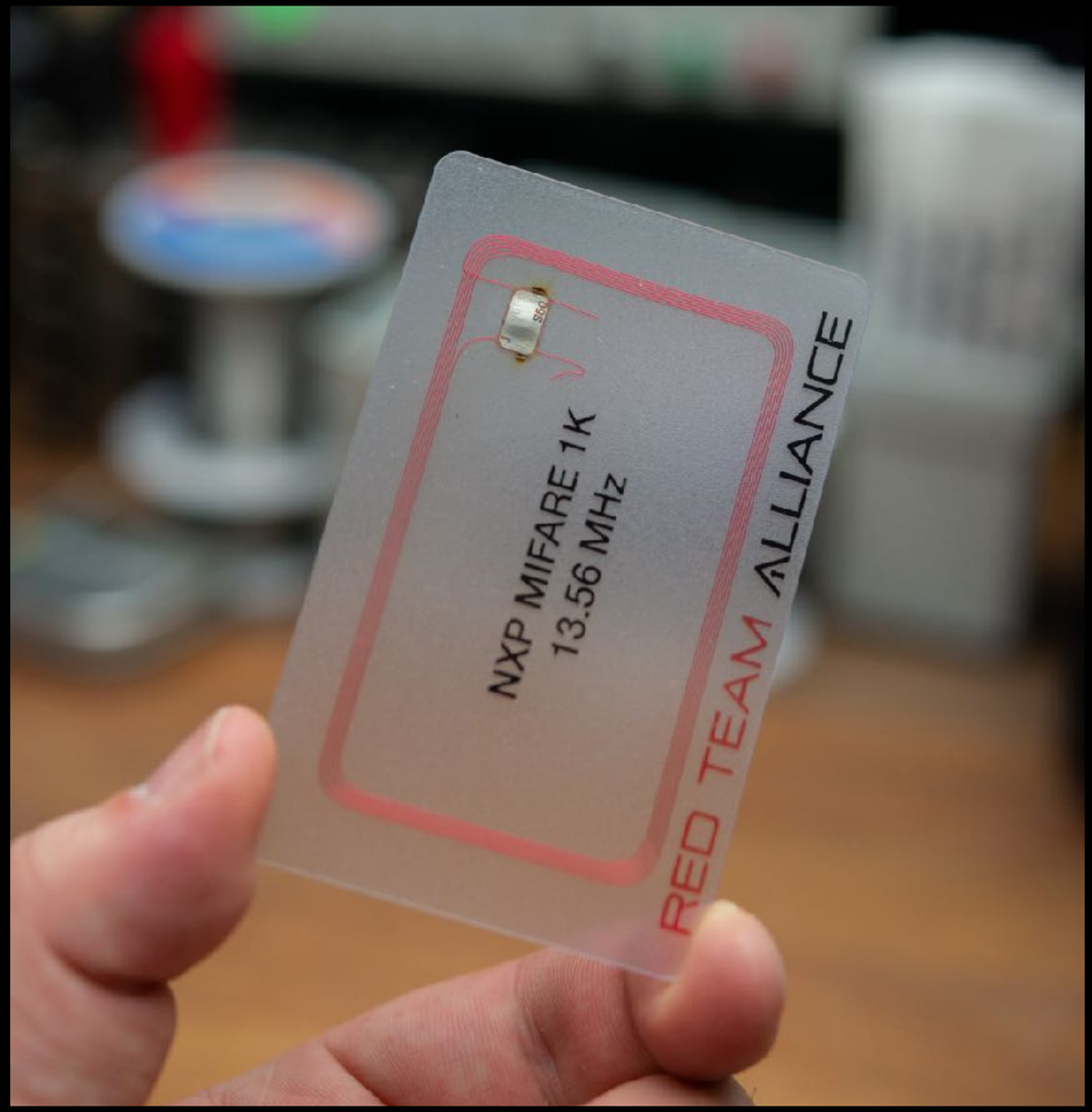
- Needs to harvest power from the active device
- Rectifies and filters the carrier frequency
- Can in some cases harvest energy to power a little bit of additional electonics

Theory and communications

- Very similar to a transformer without the iron core
- Communication from reader to tag is modulated onto the carrier frequency (ASK, Amplitude Shift Keying)
- Communications back to the reader from the tag is done by varying the load impedance

What is a tag?
NFC tag

- "Unique" ID
- Memory
- Crypto stuff (sometimes)
- Memory can be formatted in different ways
- NDEF (NFC Data Exchange Format)



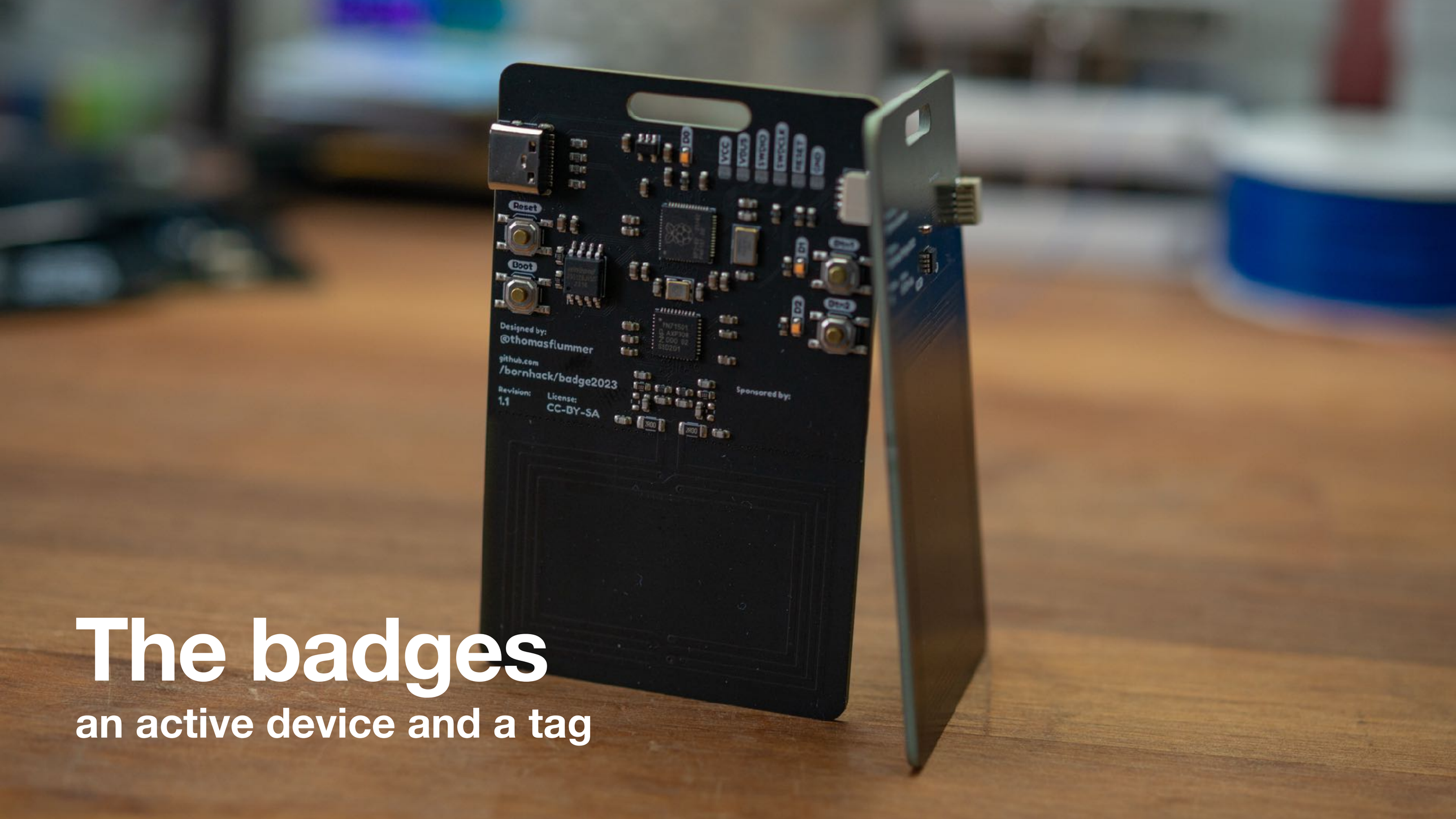
What is a "reader"

The active part

- Sort of a gateway for an application to access a tag
- Can send commands to and receive replies from a tag

An active device doing tag emulation

- This is tricky, and where it might get interesting
- Acts as if it was a tag
  - What a phone is doing when used as a payment card
- Might be able to fake being a specific tag with protected data on it



The badges

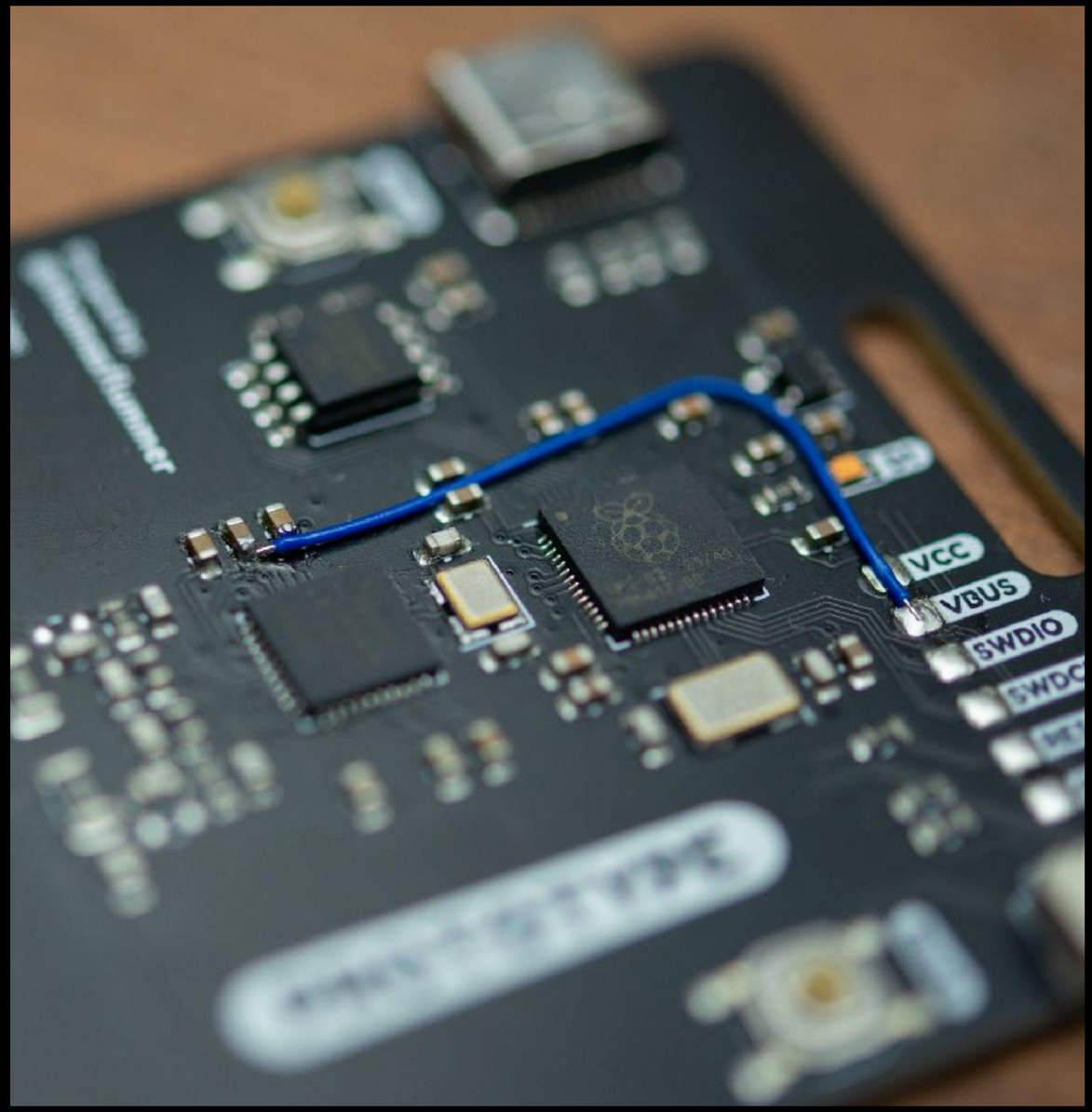
an active device and a tag

Some of the goals

- Use without a battery
  - Light
  - Easily wearable
- Functionality after the event (some sort of tool)

The electronic design

- Designing antennas/coils
- Finding parts
- Making prototypes
  - Fixing mistakes
  - Testing antenna variations



Hand assembling prototypes

SMD assembly the easy way

- The prototypes was hand assembled using a solder paste stencil, tweezers and a hotplate.
- Personally I really like this method, as it's not so stressfull, and you can fix small mistakes before you do the actual soldering, which makes it much easier.
- There is a workshop **Friday at 12:00** by Theo Borm using this same technique which is highly recommended

Hardware design in KiCad 7

- All the hardware design is done in KiCad 7 and available at:
  - https://github.com/bornhack/badge2023
- There is a workshop on **Sunday at 16:00** by Theo Borm if you are interested in learning KiCad

Tag

Passive device, very few components

- Uses the **NTAG I²C Plus** from NXP (NT3H2211)
- ISO/IEC 14443 Part 2 and 3 compliant
- Has a unique 7 bit number/ID
- 2k of memory
- Also have an I2C interface, that allows manipulating the same memory that is accessable from the RF side
- There are some protective settings to limit what can be done
- **WARNING!** it is possible to lock yourself out

Reader

Active device, USB powered

- Uses NXP **PN7150** for NFC
- Raspberry Pi **RP2040** for USB and applications
- 16MB SPI flash for program and data files
- You can connect the tag badge, and communicate with the NT3H2211 chip via I²C, read/write content of memory
- Has 2 user buttons and three LEDs
- There are also pads for SWD debug


right
RESET
START/STOP
POWER
LIGHT

Programming the reader

CircuitPython

- The badges comes with CircuitPython preloaded
- There is a simple example that talks to the reader chip
- Esmil has been working on a CP library for the PN7150
- You can also change the memory/registers via I^2C on the tag with the included cable
- The cable connection follows the Sparkfun Qwiic/Adafruit StemmaQT standard

Programming the reader

Arduino IDE/framework

- PN7150 library by ElectronicCats
  - https://github.com/ElectronicCats/ElectronicCats-PN7150
- The library includes some examples, eg. a reader, that gets the ID number of a tag.
- You will also need to install an RP2040 board package in Arduino IDE

Ideas for projects

We might look into some of these in the badge team during camp

- USB UART to NCI (NFC Controller Interface) Bridge
  - Would allow the reader badge to act as a generic NFC reader, eg. in Linux and make it compatible with standard software
- USB Keyboard emulation
  - Take the ID number of a tag, and "type" it as if it was entered as keystrokes

Changing the content on the tag

via the radio interface

- Mobile (Android/iOS) apps
  - NFC Tools
  - NXP TagWriter
- Desktop apps (needs NFC reader connected via USB)
  - NFC Tools

Other tools

Proxmark 3 RDV 4.01

- Swiss army knife of RFID
- Needs to be connected to a computer or similar

Other tools

Flipper Zero

- Handheld/portable
- Battery powered
- Gamified physical security tool/ toy

Other tools

Chameleon

- Emulate different NFC tags
- Setup via computer
- Selfcontained use

RFID Test / blocking cards

- If you want to find out what technology a reader is using (LF or HF), there are some clever test cards, that uses the power harvesting and carrier frequency to light up LEDs
- To protect against scan attacts, blocking cards exist

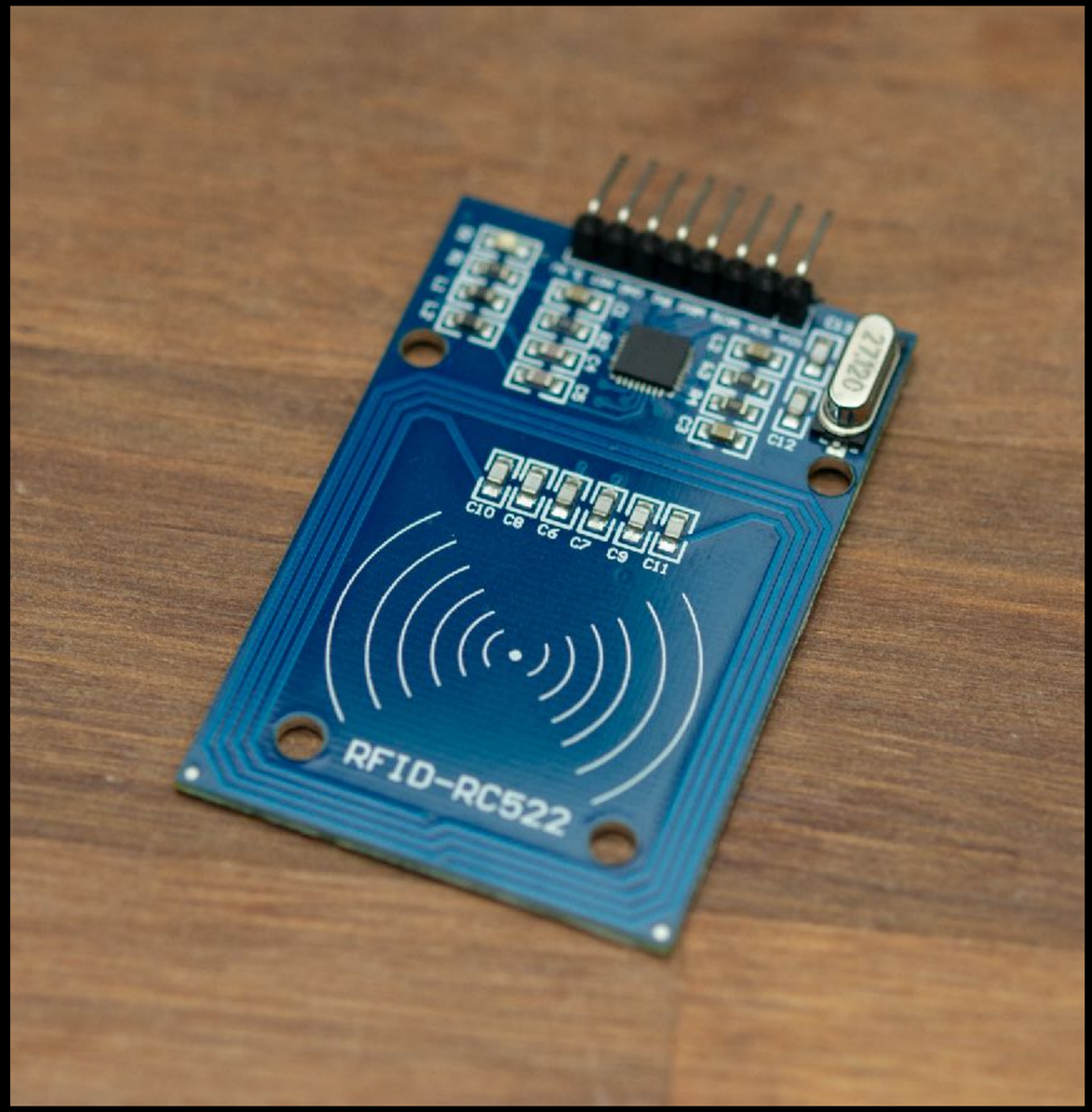
Game / token hunt

- The tag has a "unique" ID
- We have made some reader stations, that will provide some interactivity during camp
- Please don't take apart the stations

Make your own station

eg. in your village

- This is the reader that is used inside the stations
- Some of these are available in hardware hacking area close to the cabin
- Fairly easy to connect to Raspberry Pi or similar via SPI



Hardware design and code is on Github

github.com/bornhack/badge2023

- Please make pull requests to the badge repository
- We use a slightly different model, with different things in branches, feel free to add stuff you make as a separate branch
- PR with changes to the README.md that includes links to what you have made are also very welcome
- Hardware design is in KiCad 7, and you will need the current stable or a nightly build to open the files, KiCad 6 or older will complain a little.