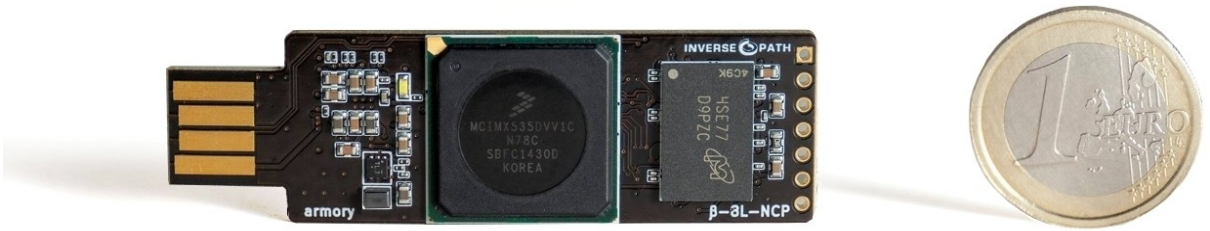
INVERSE PATH

Forging the USB armory

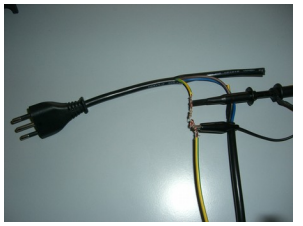
Andrea Barisani

<andrea@inversepath.com>

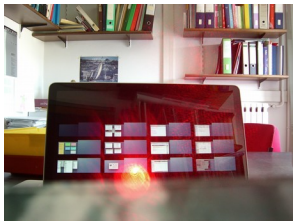


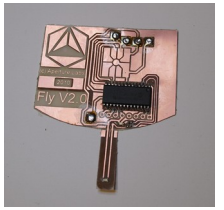
INVERSE PATH

2007: Unusual Car Navigation Tricks
Injecting RDS-TMC Traffic Information Signals



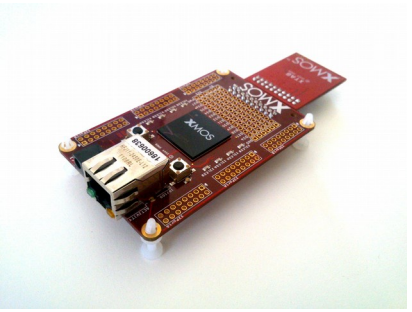
2009: Sniff Keystrokes With Lasers/Voltmeters
Side Channel Attacks Using Optical Sampling Of Mechanical Energy And Power Line Leakage

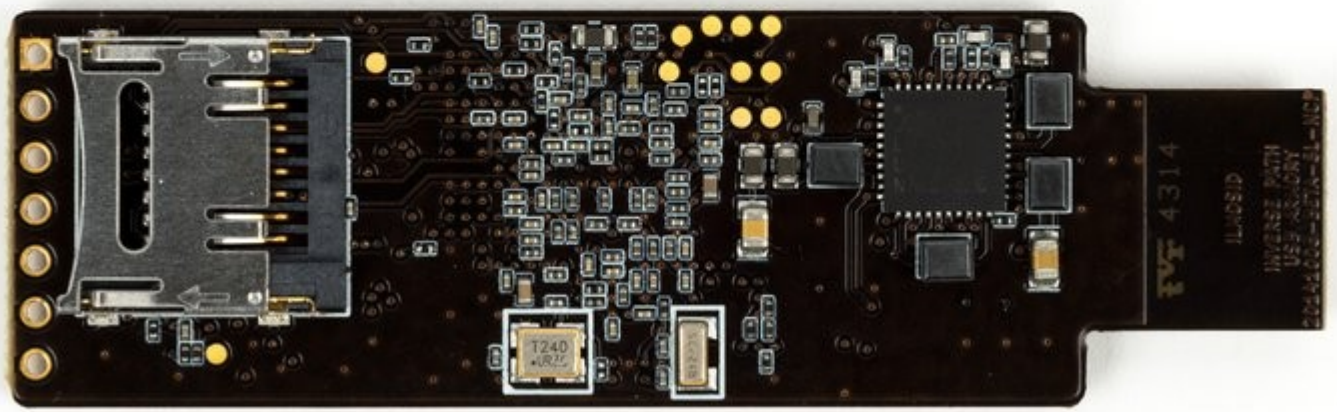




2011: Chip & PIN is definitely broken
Credit card skimming and PIN harvesting in an EMV world

2013: Fully arbitrary 802.3 packet injection
Maximizing the Ethernet attack surface





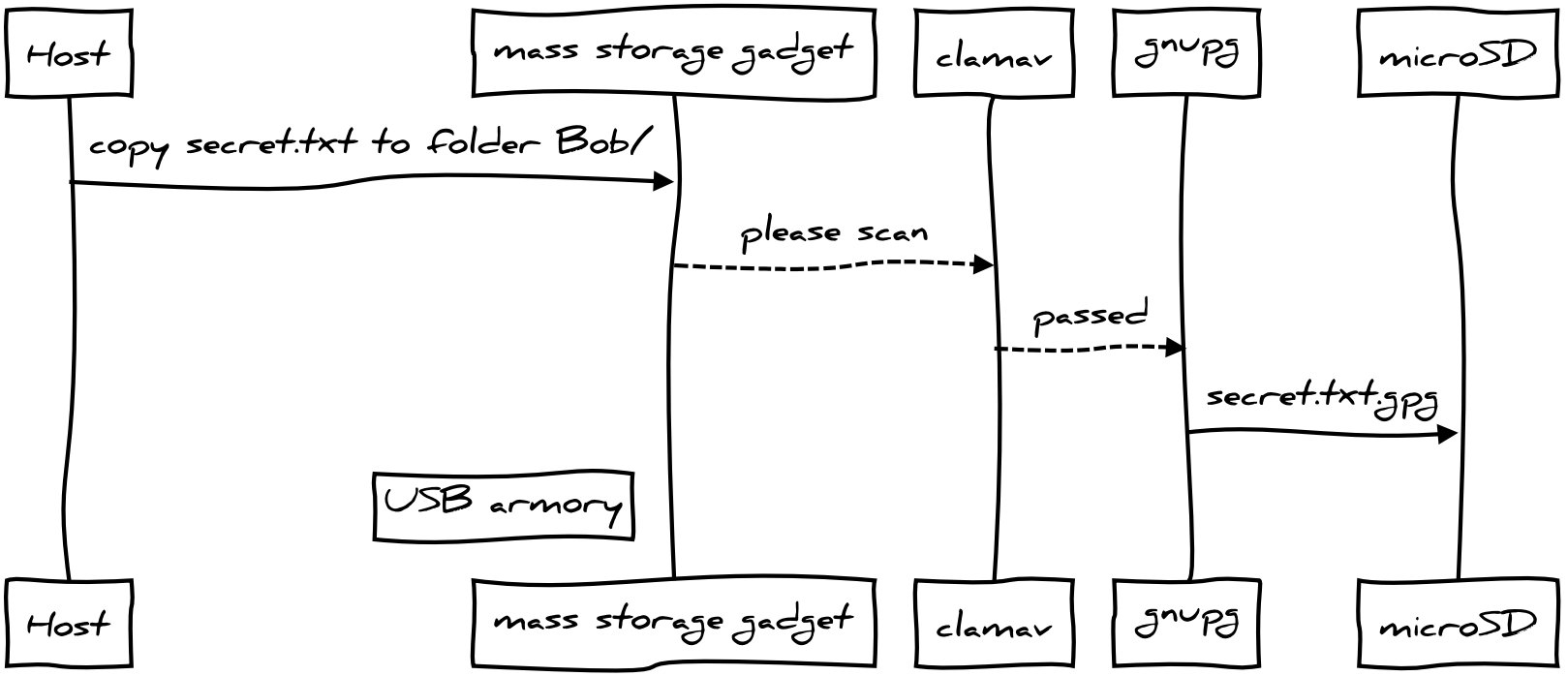
INVERSE PATH

Designed for personal security applications

- mass storage device with advanced features such as automatic encryption, virus scanning, host authentication and data self-destruct
- OpenSSH client and agent for untrusted hosts (kiosk)
- router for end-to-end VPN tunneling, Tor
- password manager with integrated web server
- electronic wallet (e.g. pocket Bitcoin wallet)
- authentication token
- portable penetration testing platform
- low level USB security testing

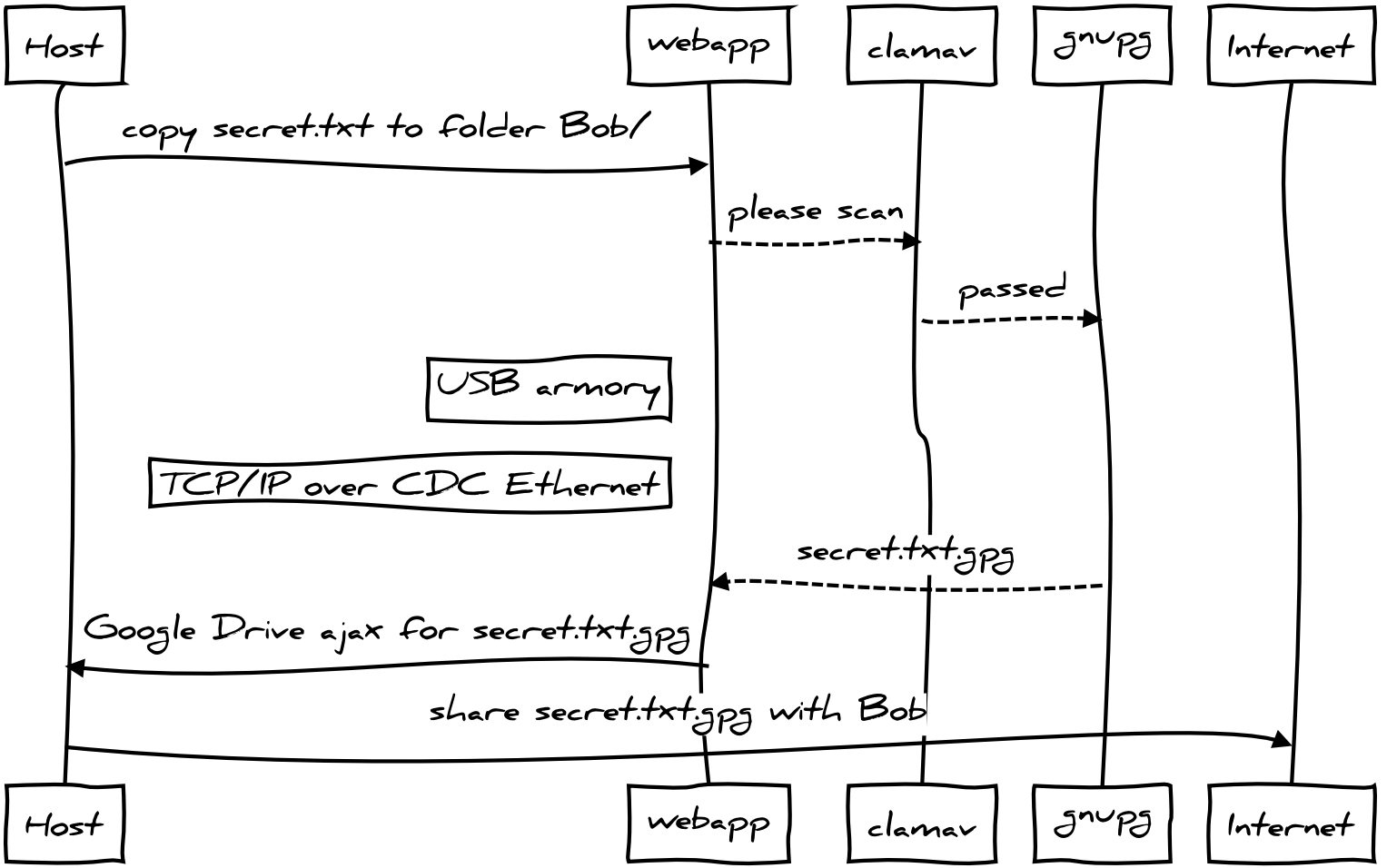
INVERSE PATH

enhanced mass storage



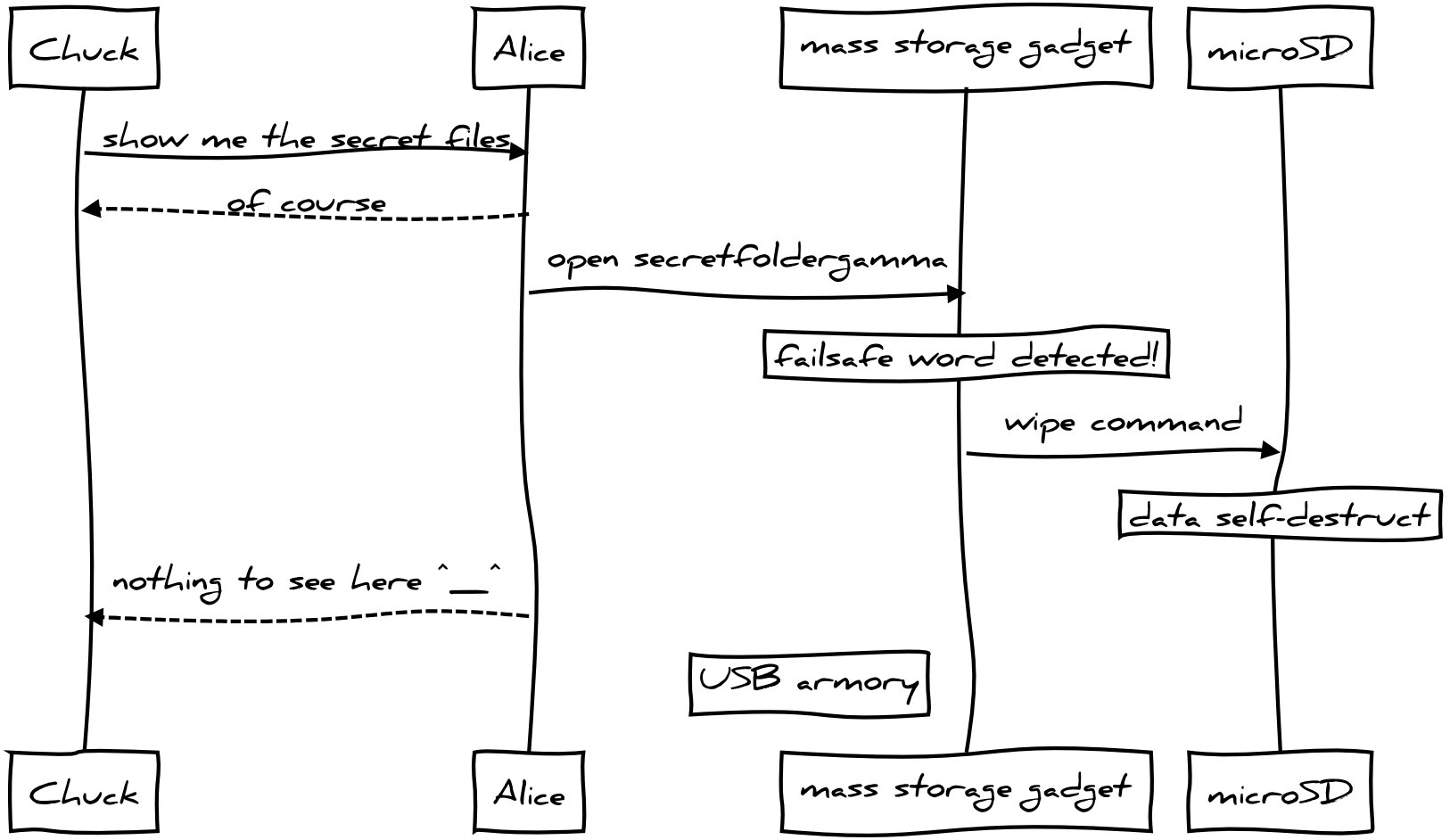
INVERSE PATH

enhanced mass storage



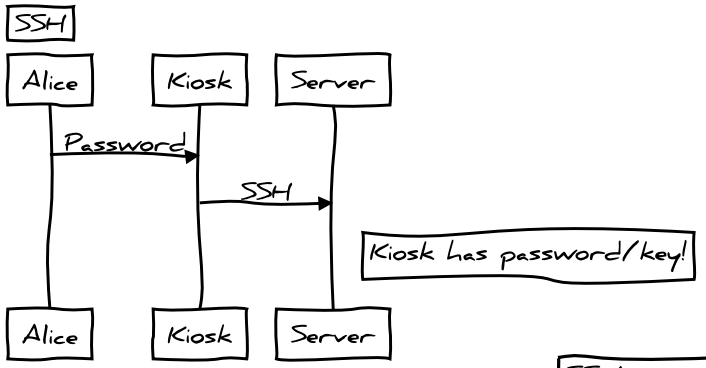
INVERSE PATH

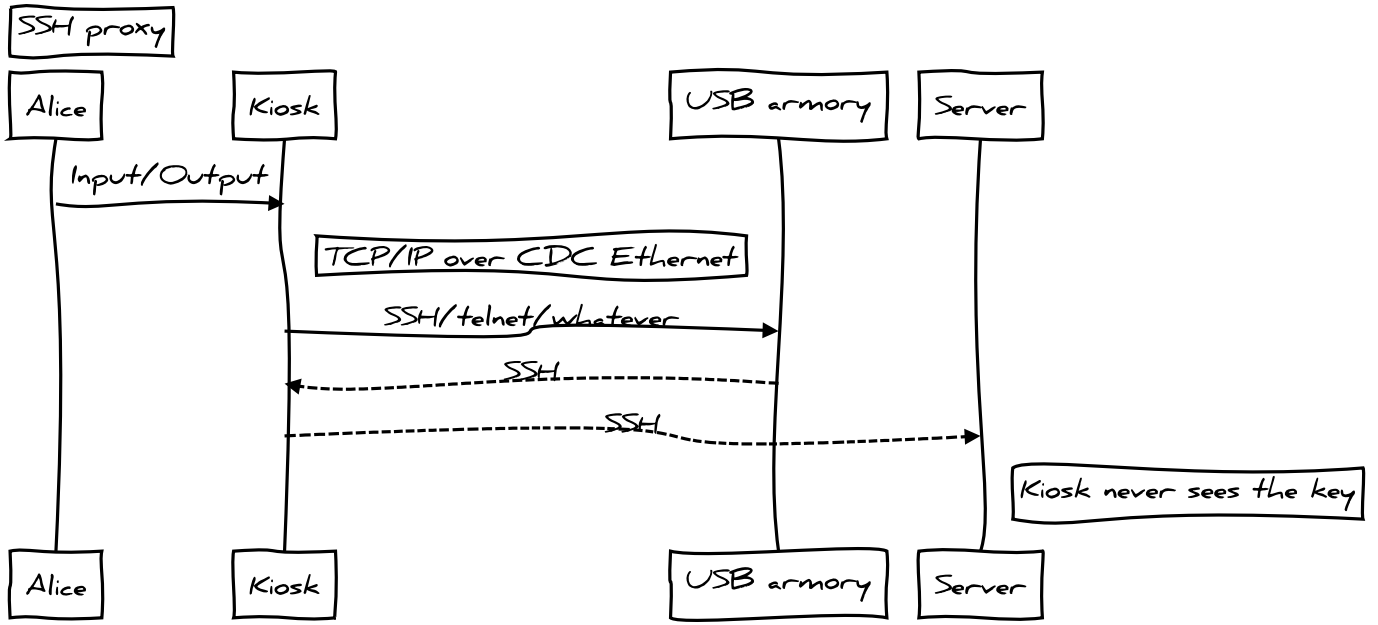
enhanced mass storage

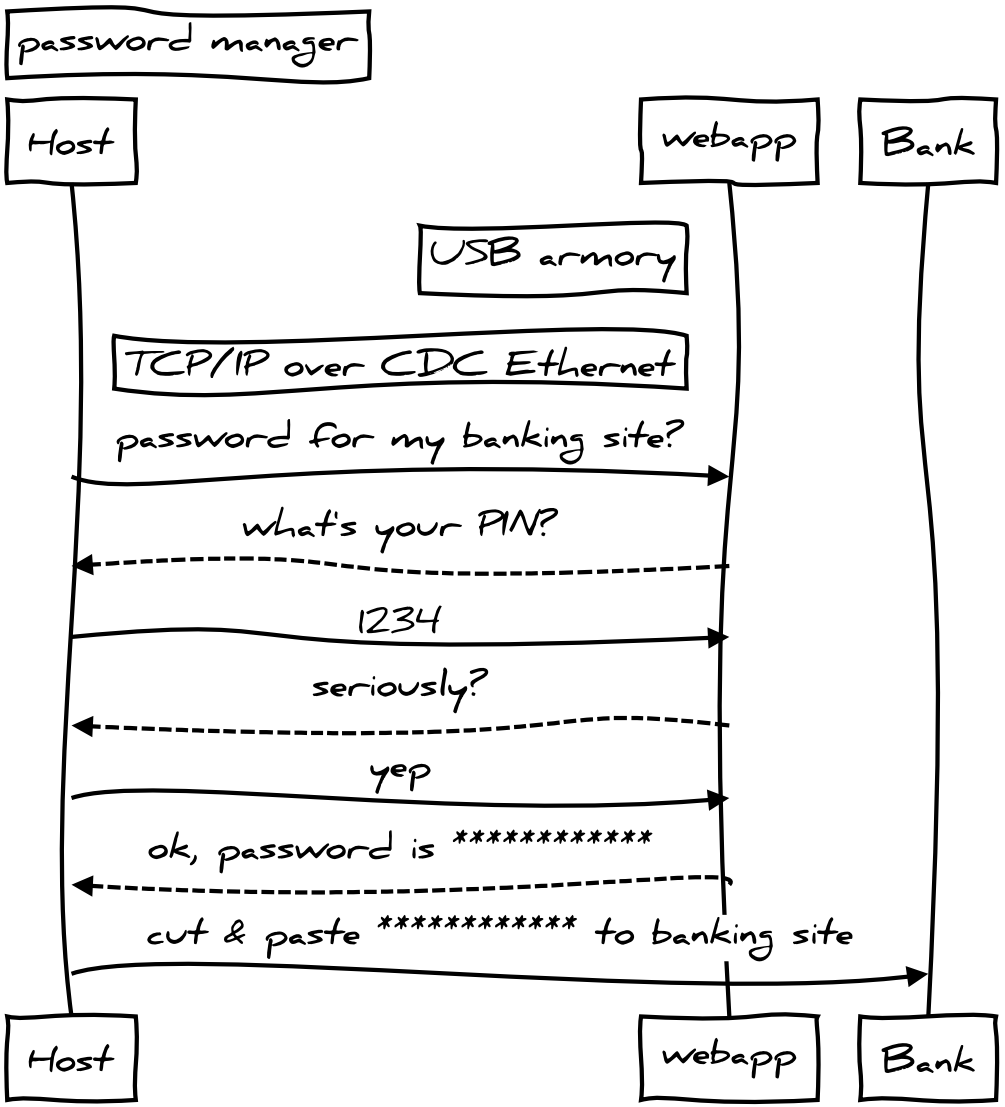


INVERSE PATH

SSH proxy





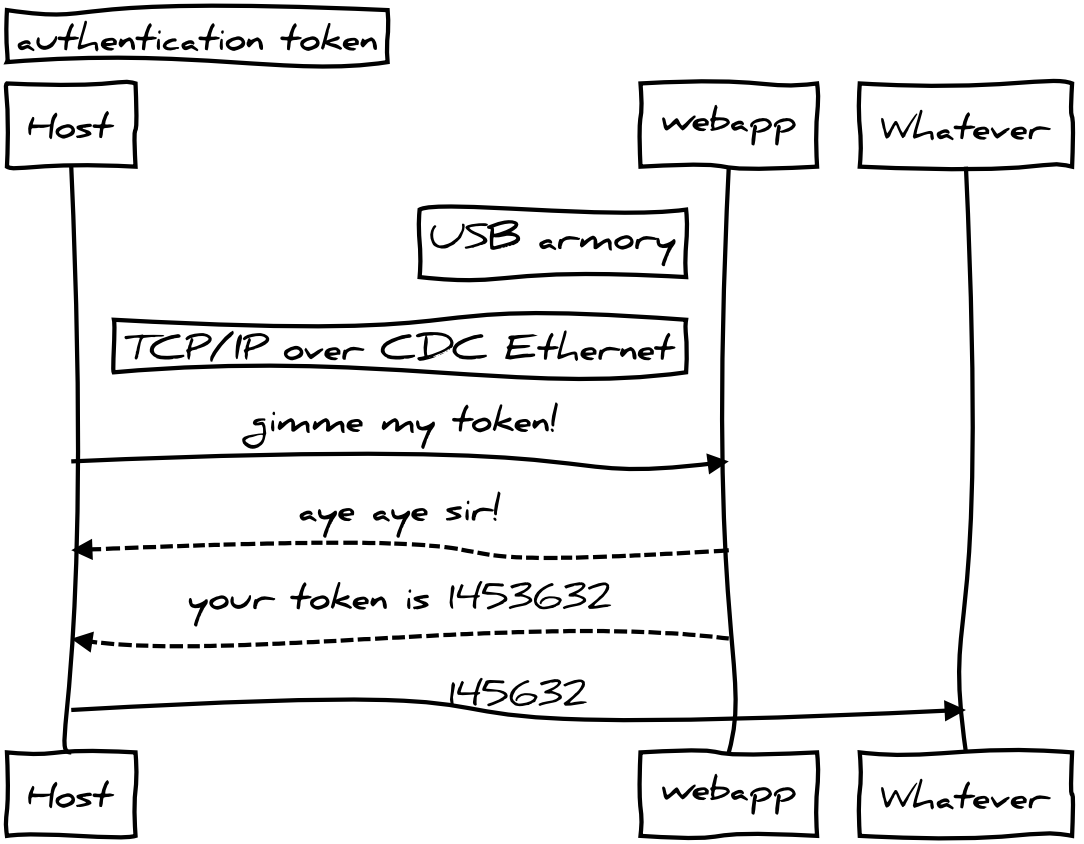


password manager

**trivial example, better options planned*

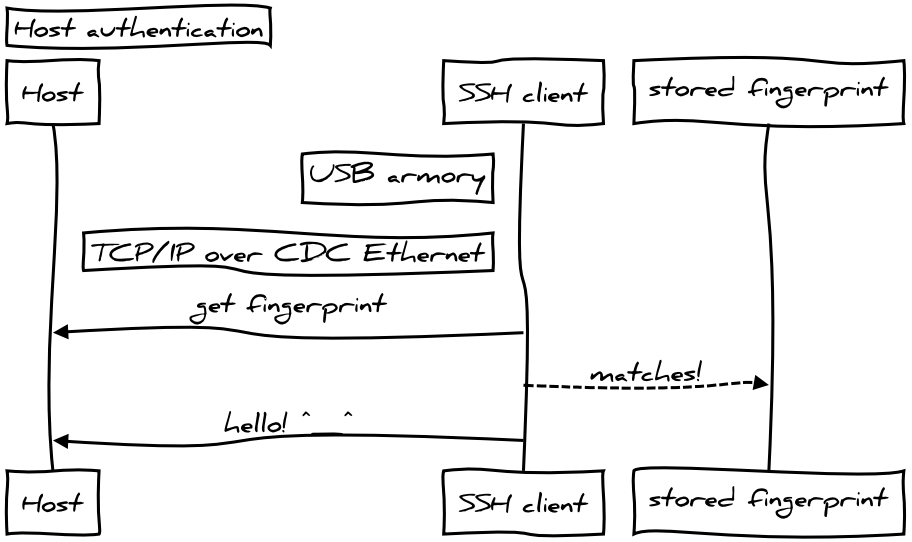
INVERSE PATH

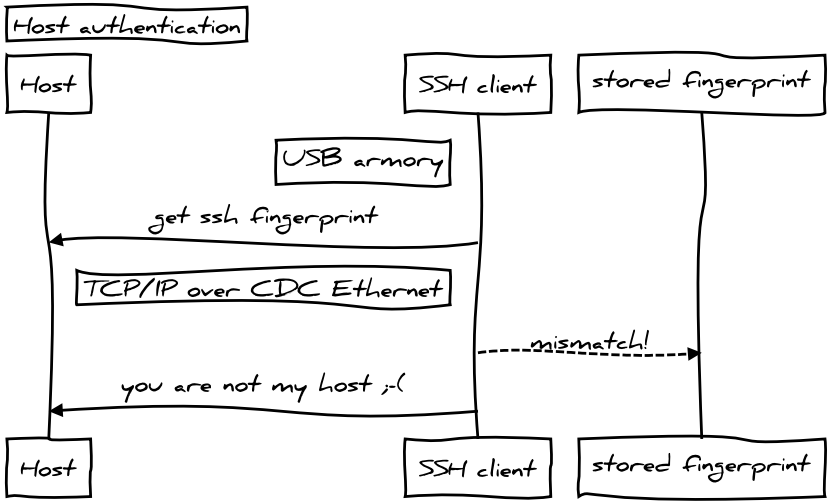
authentication token



INVERSE PATH

USB device authenticates host





INVERSE PATH

Design goals

Compact USB powered device

Fast CPU and generous RAM

Secure boot

Standard connectivity over USB

Familiar developing/execution environment

Open design

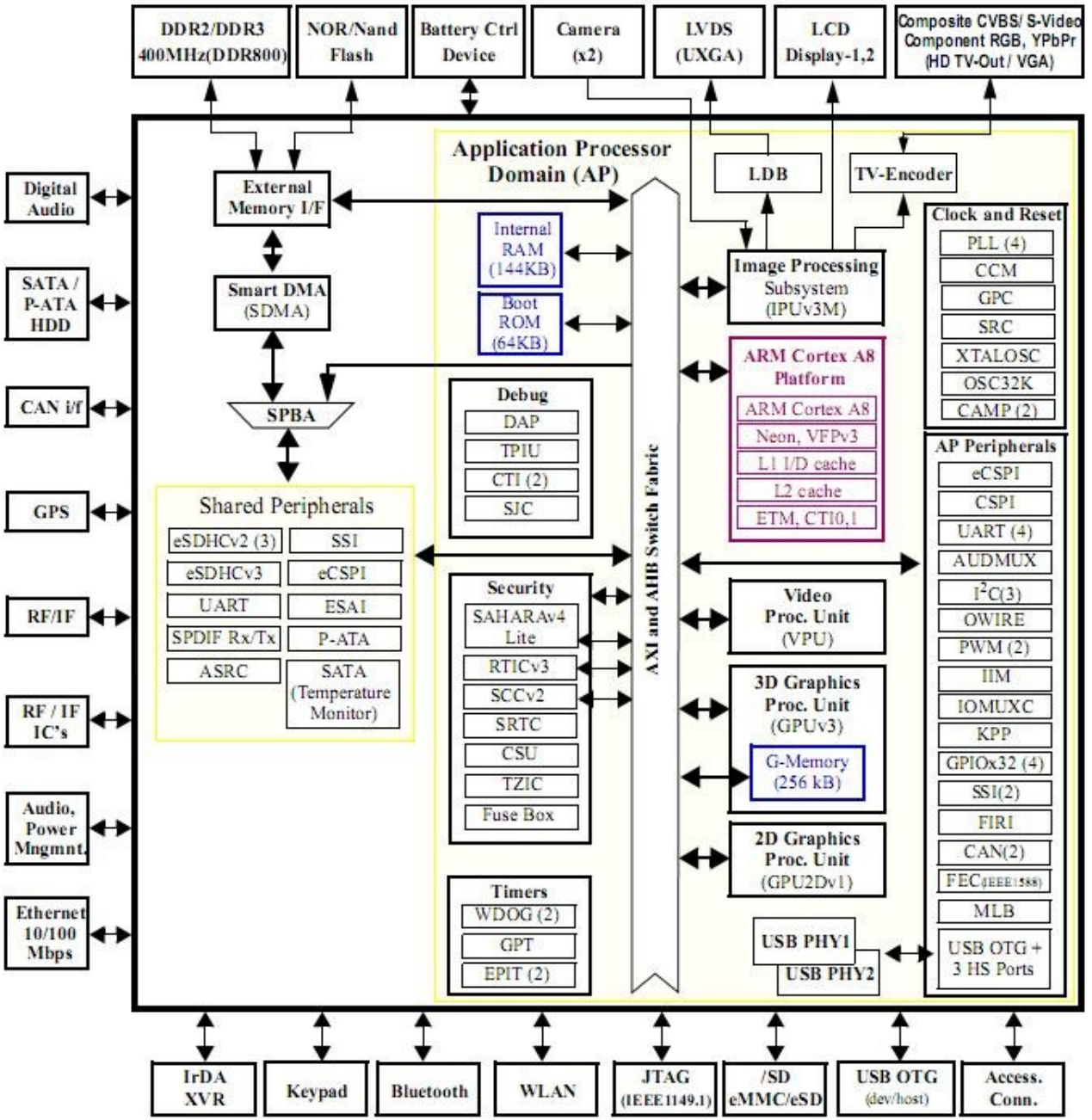
INVERSE PATH

Selecting the System on Chip (SoC)

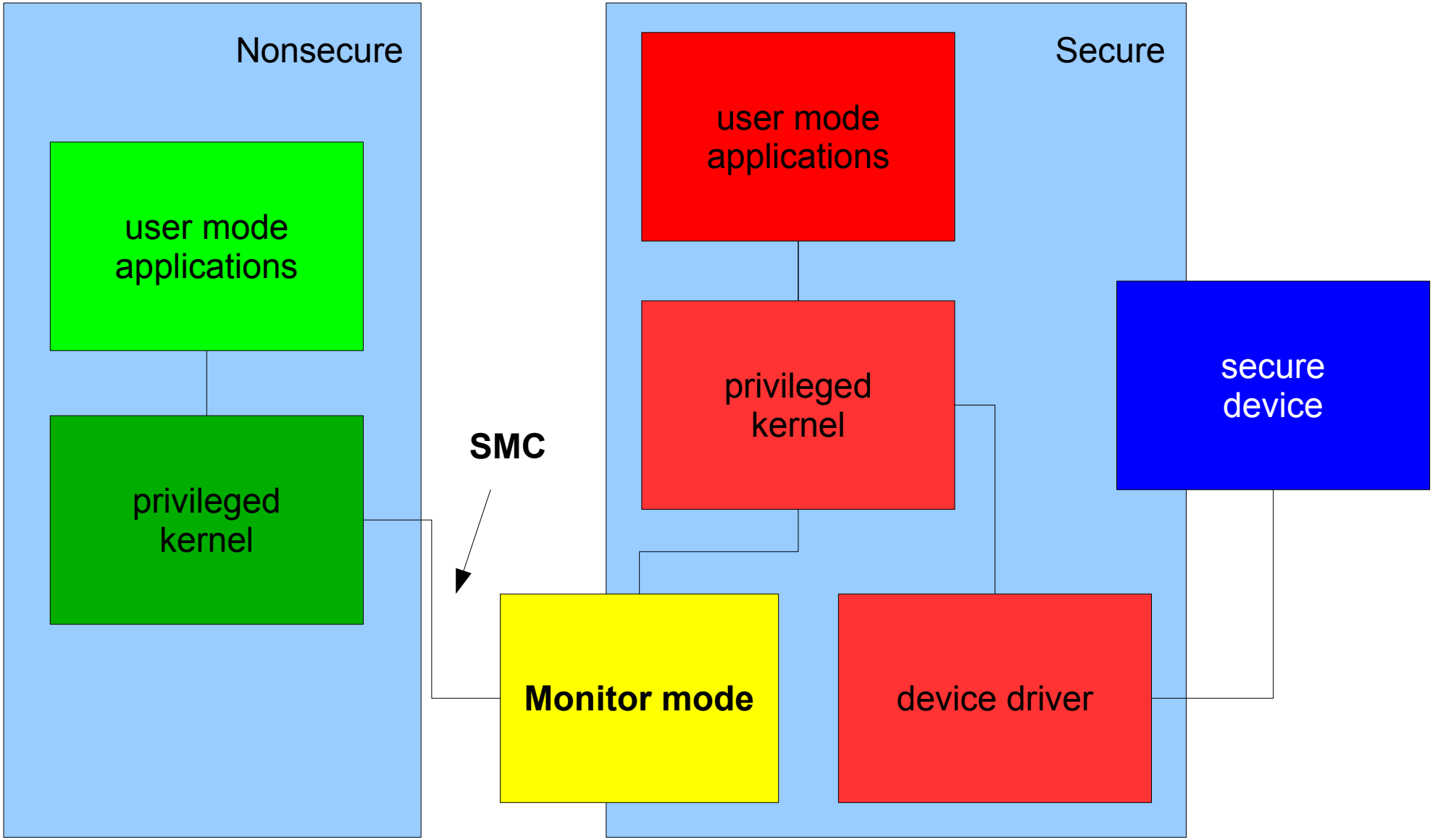
Freescale i.MX53

- ARM® Cortex™-A8 800-1200 Mhz
- almost all datasheets/manuals are public (no NDA required)
- Freescale datasheets are “ok” (far better than other vendors)
- ARM® TrustZone®, secure boot + storage + RAM
- detailed power consumption guide available
- excellent native support (Android, Debian, Ubuntu, FreeBSD)
- good stock and production support guarantee

INVERSE PATH

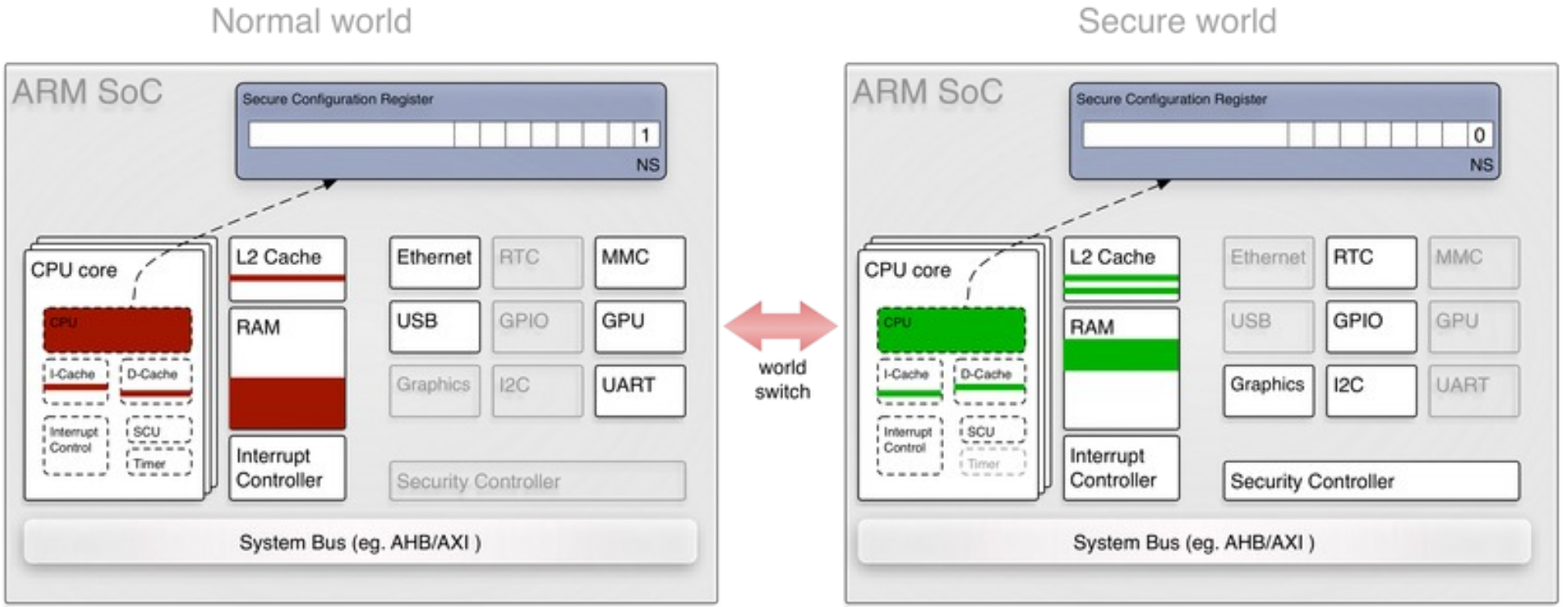


Domain Separation



INVERSE PATH

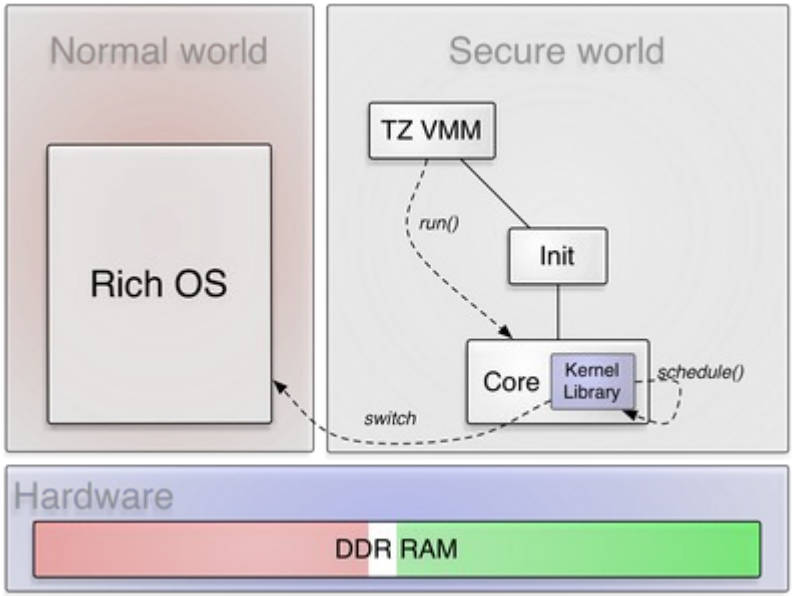
ARM® TrustZone®

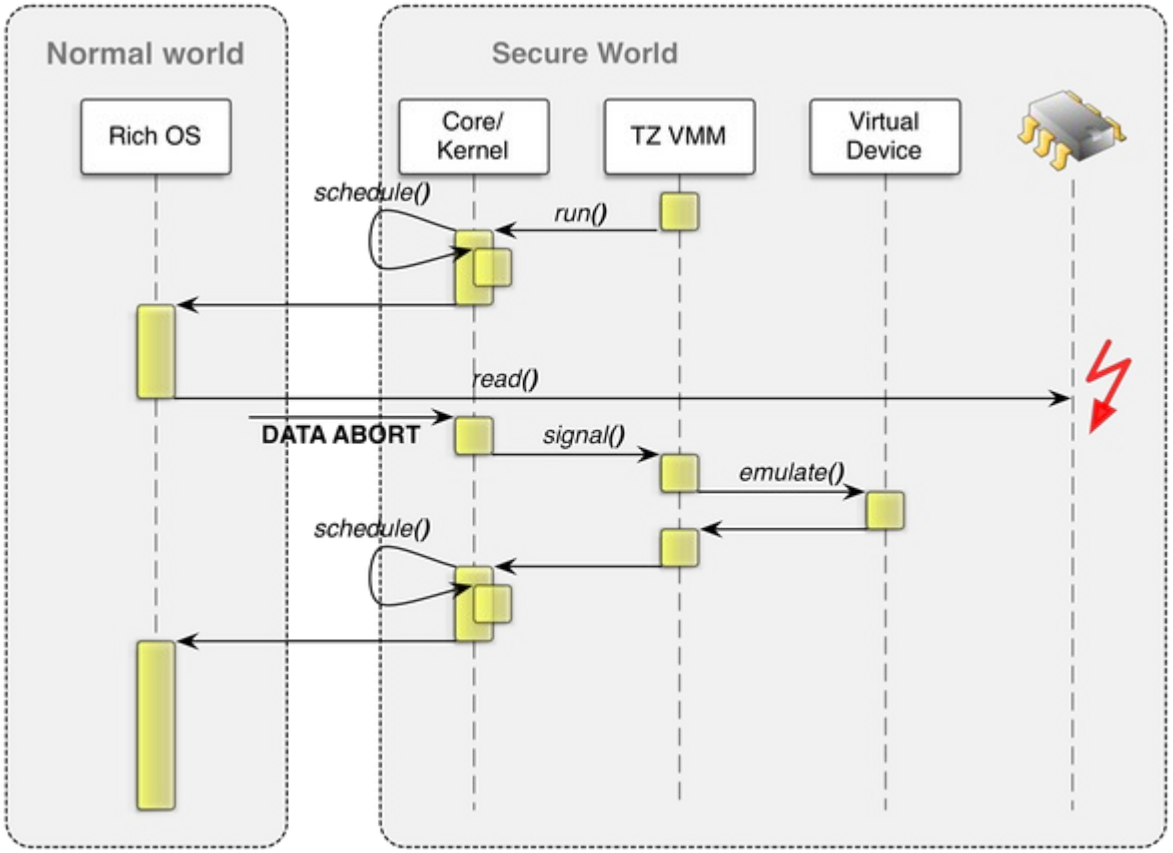


http://genode.org/documentation/articles/trustzone

INVERSE PATH

ARM® TrustZone®





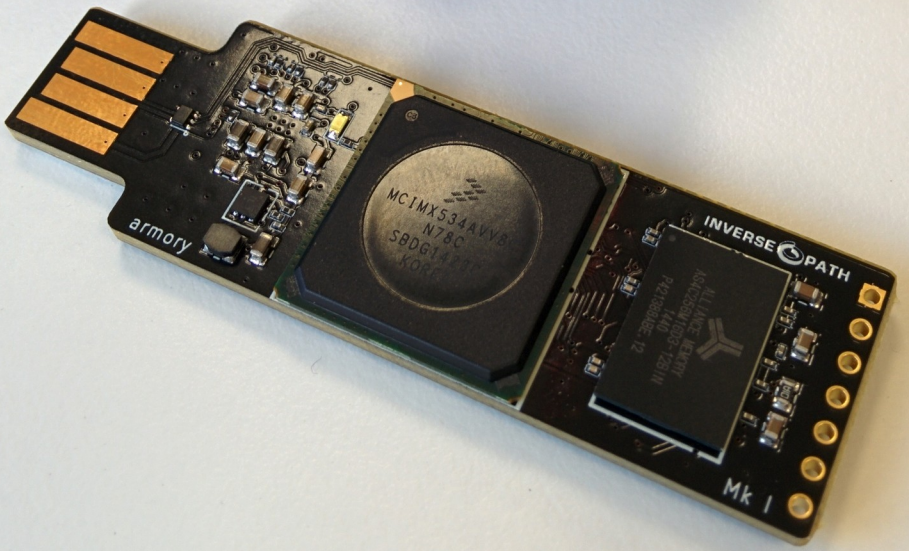
http://genode.org/documentation/articles/trustzone

INVERSE PATH

Development time-line

- 2014/03: schematics development begins (Freescale chosen)
- 2014/04: PCB layout for breakout/prototyping board
- 2014/08: alpha board order
- 2014/09: USB armory alpha board delivery & evaluation
- 2014/10: project announcement
- 2014/10: order for 7 optimized beta revisions
- 2014/11: beta boards delivery & evaluation
- 2014/11: design finalization, Mk I production candidate order
- 2014/12: Mk I delivery
- 2015/01: first batch production (1k)
- 2015/03: shipping begins!

INVERSE PATH



open source hardware

open source

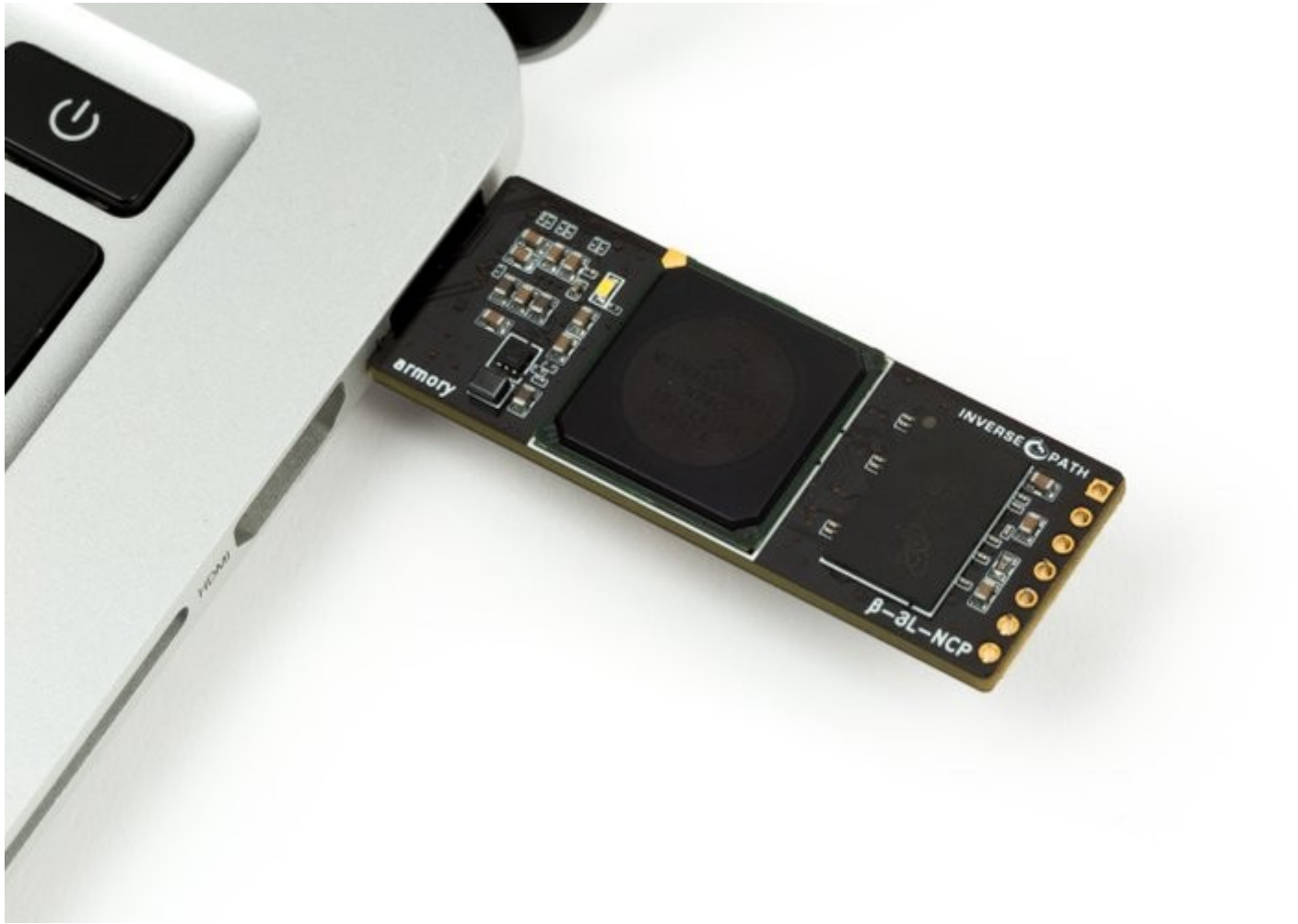
http://inversepath.com/usbarmory

Forging the USB armory

INVERSE PATH

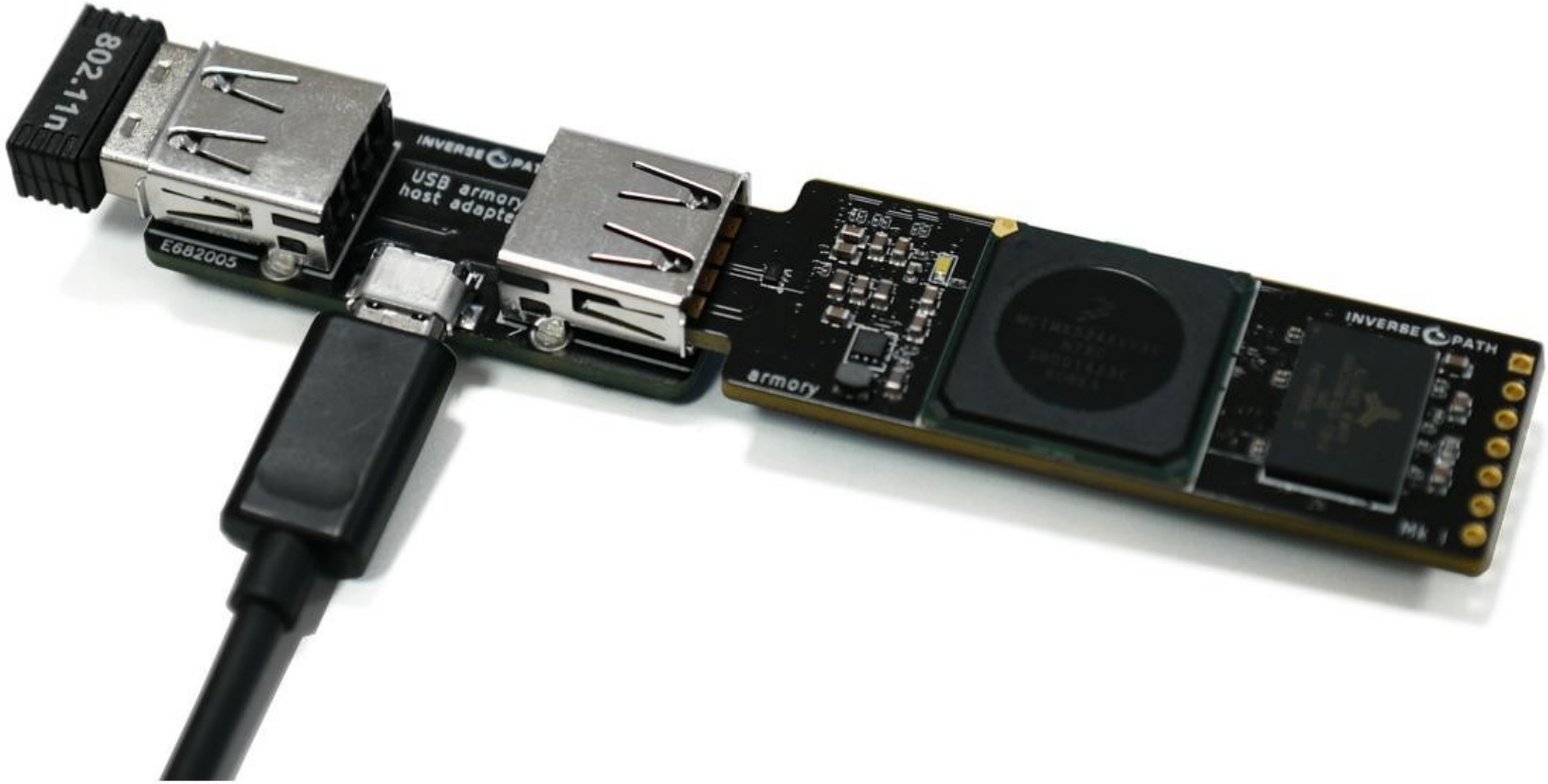
USB armory - Open source flash-drive-sized computer

- Freescale i.MX53 ARM® Cortex™-A8 800Mhz, 512MB DDR3 RAM
- USB host powered (<500 mA) device with compact form factor (65 x 19 x 6 mm)
- ARM® TrustZone®, secure boot + storage + RAM
- microSD card slot
- 5-pin breakout header with GPIOs and UART
- customizable LED, including secure mode detection
- excellent native support (Debian, Ubuntu, Arch Linux ARM)
- USB device emulation (CDC Ethernet, mass storage, HID, etc.)
- Open Hardware & Software



device mode

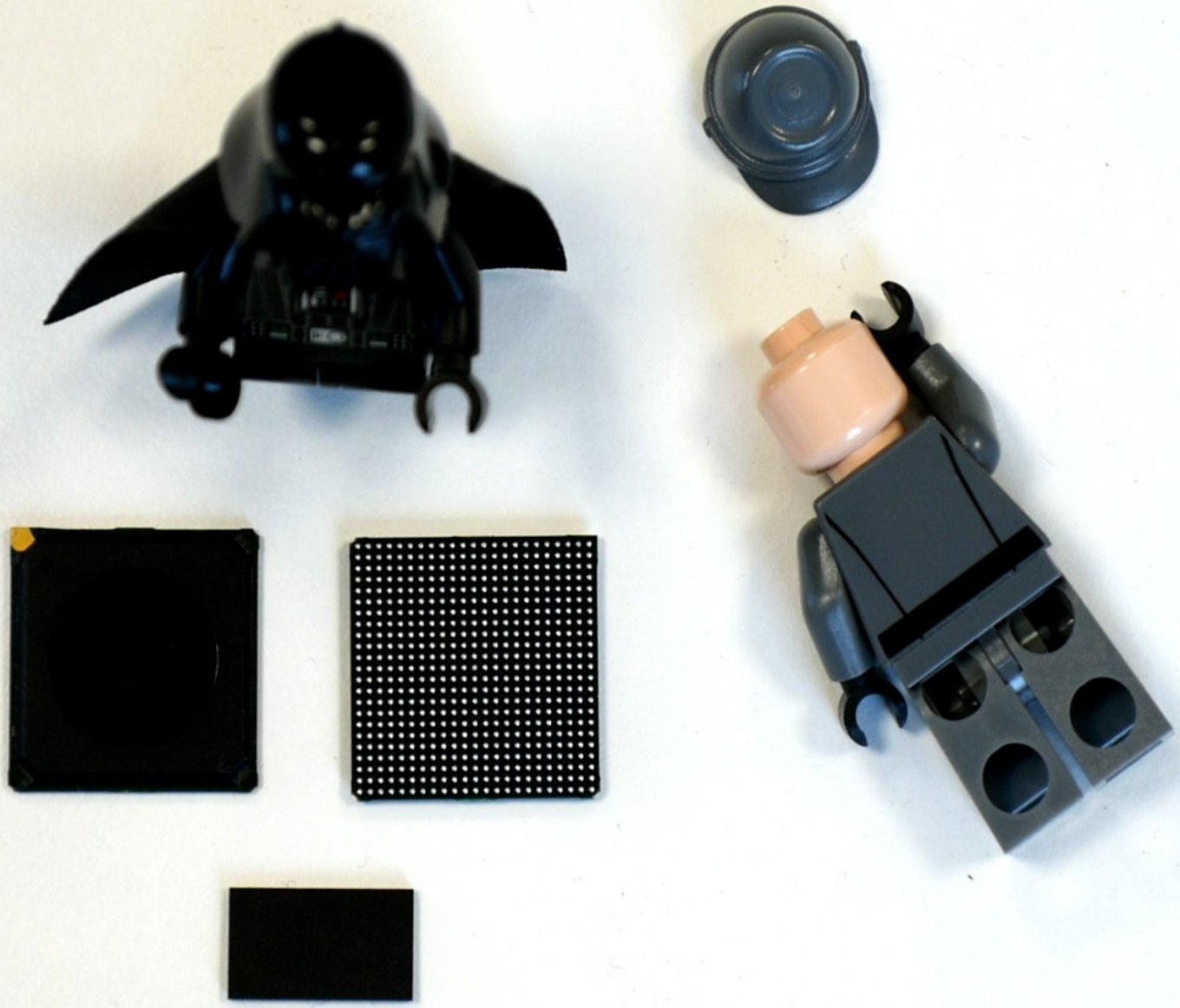
INVERSE PATH



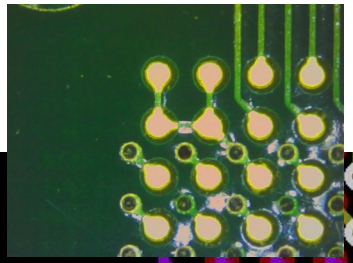
INVERSE PATH

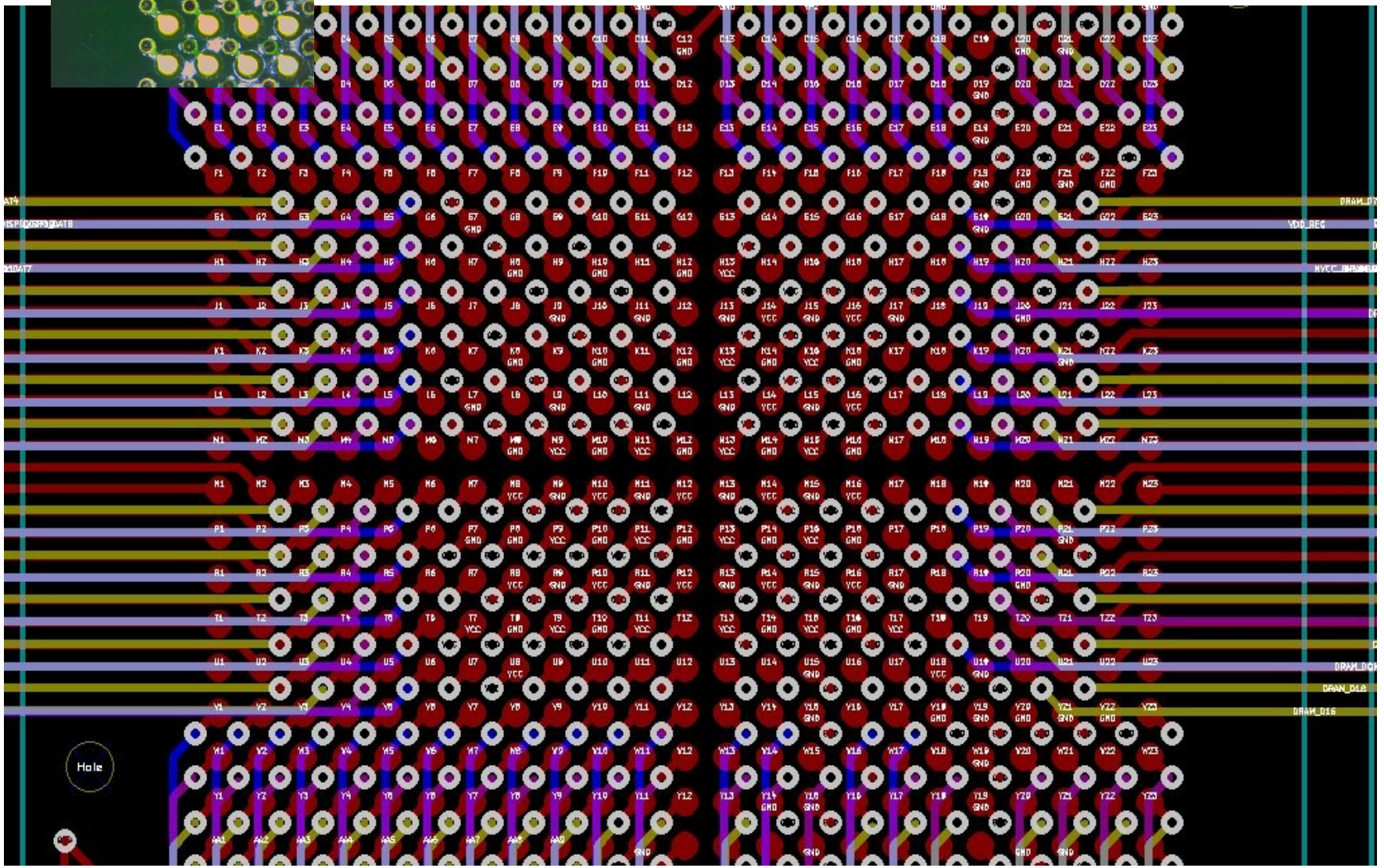
inversepath.com/usbarmory

INVERSE PATH

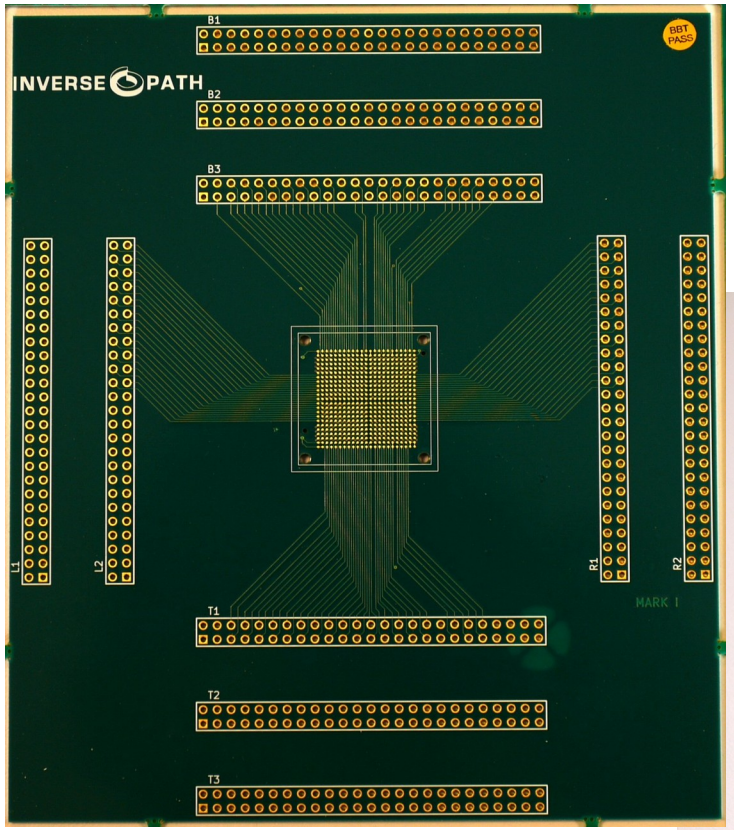


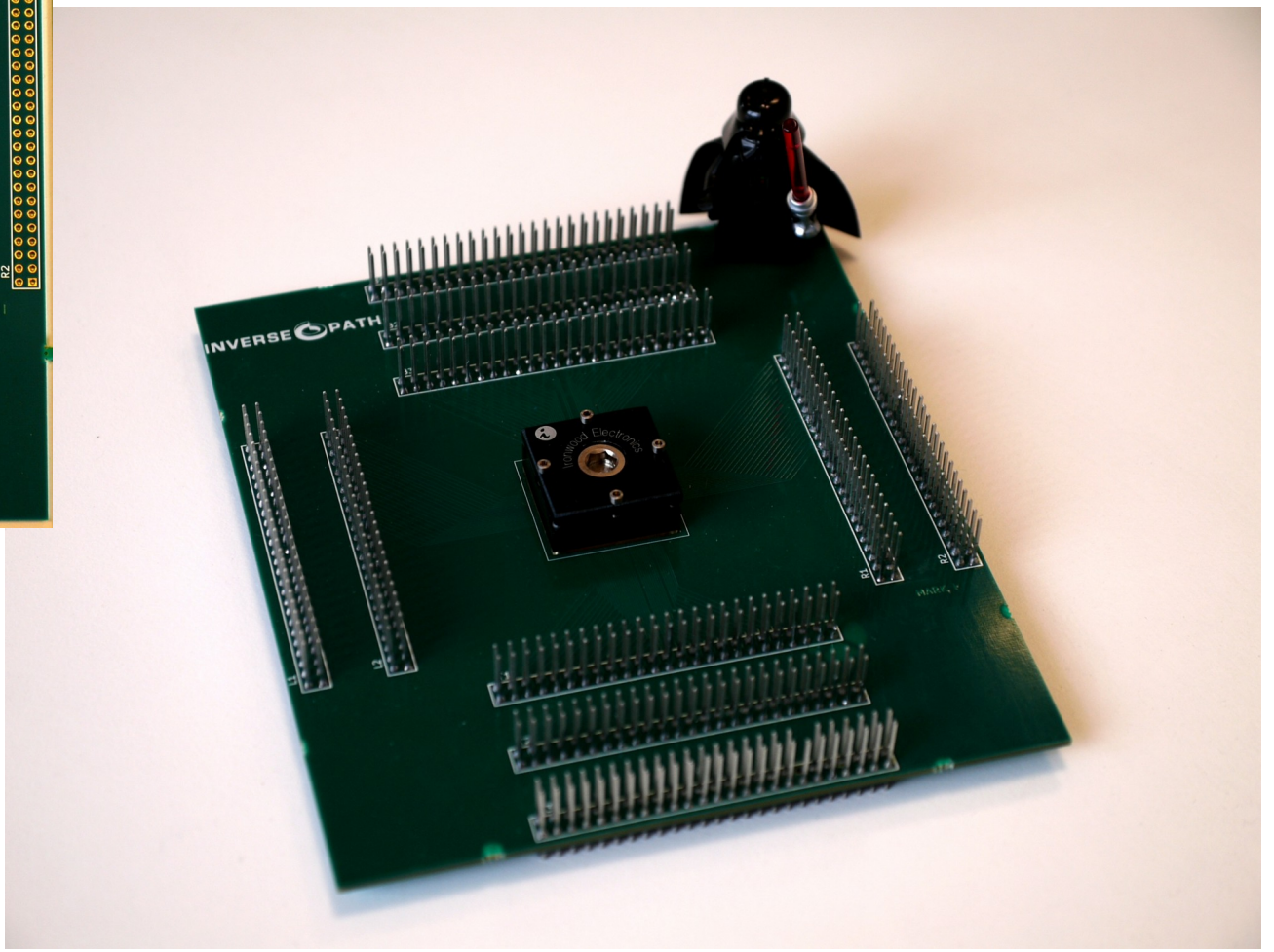
INVERSE PATH



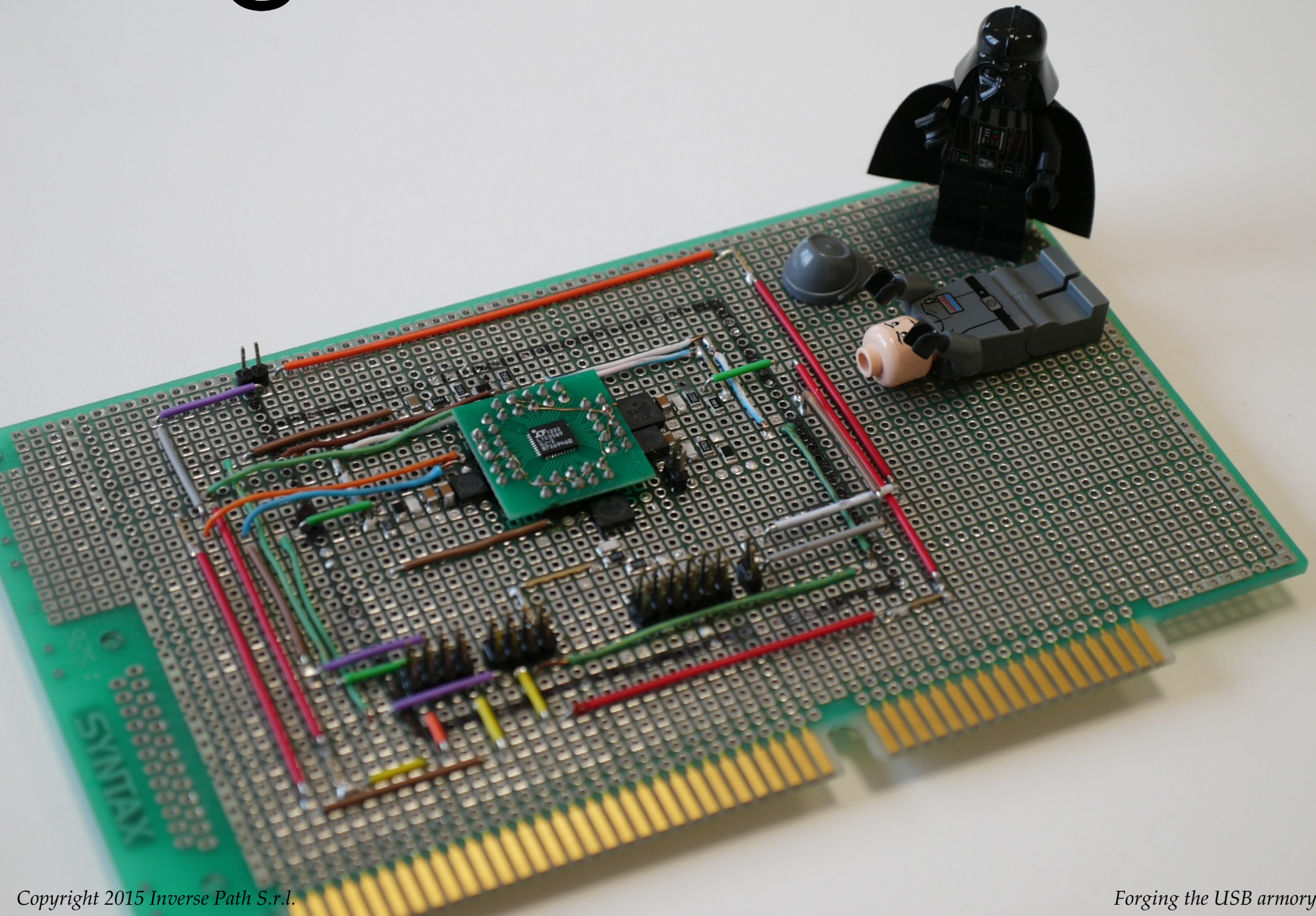


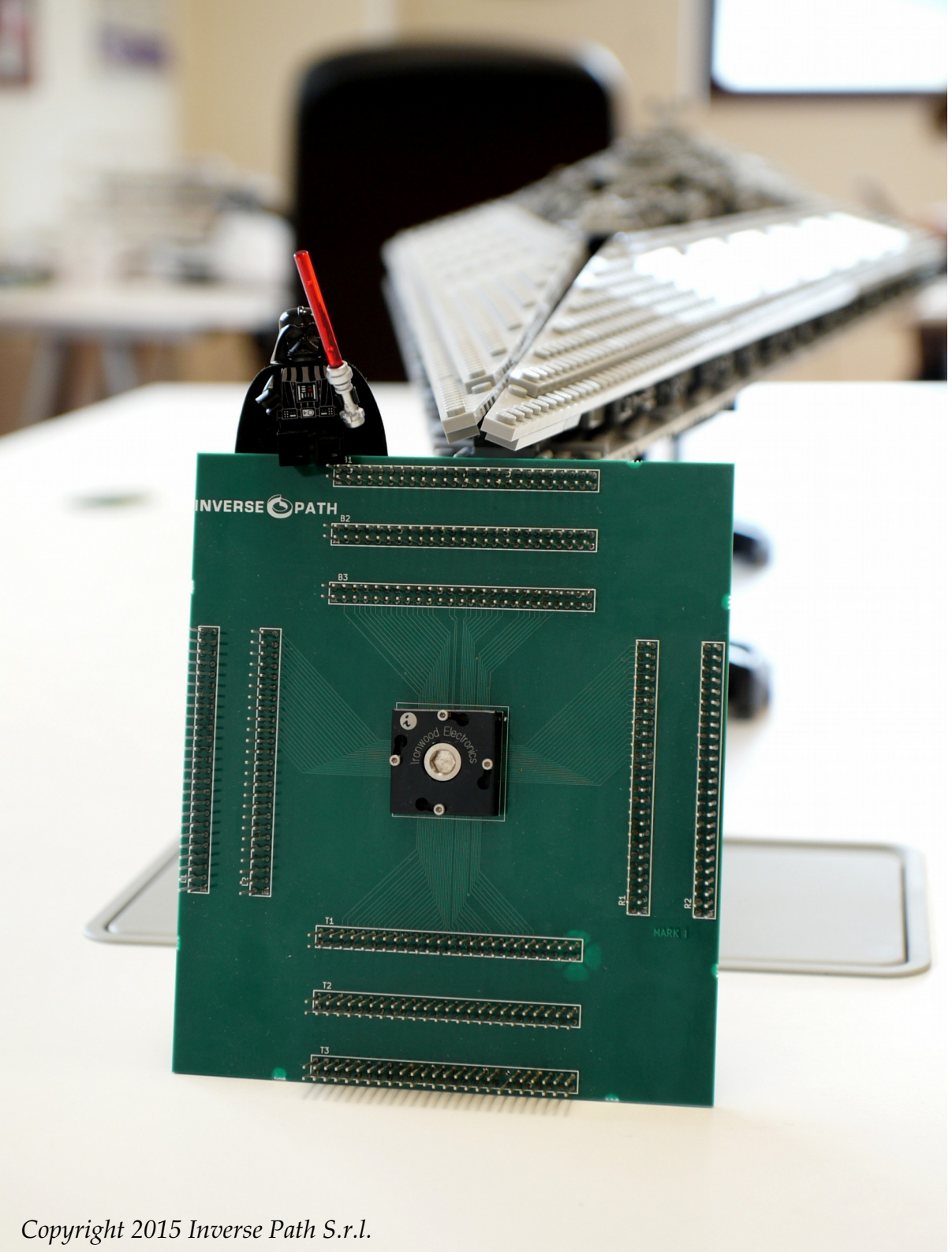
INVERSE PATH





INVERSE PATH

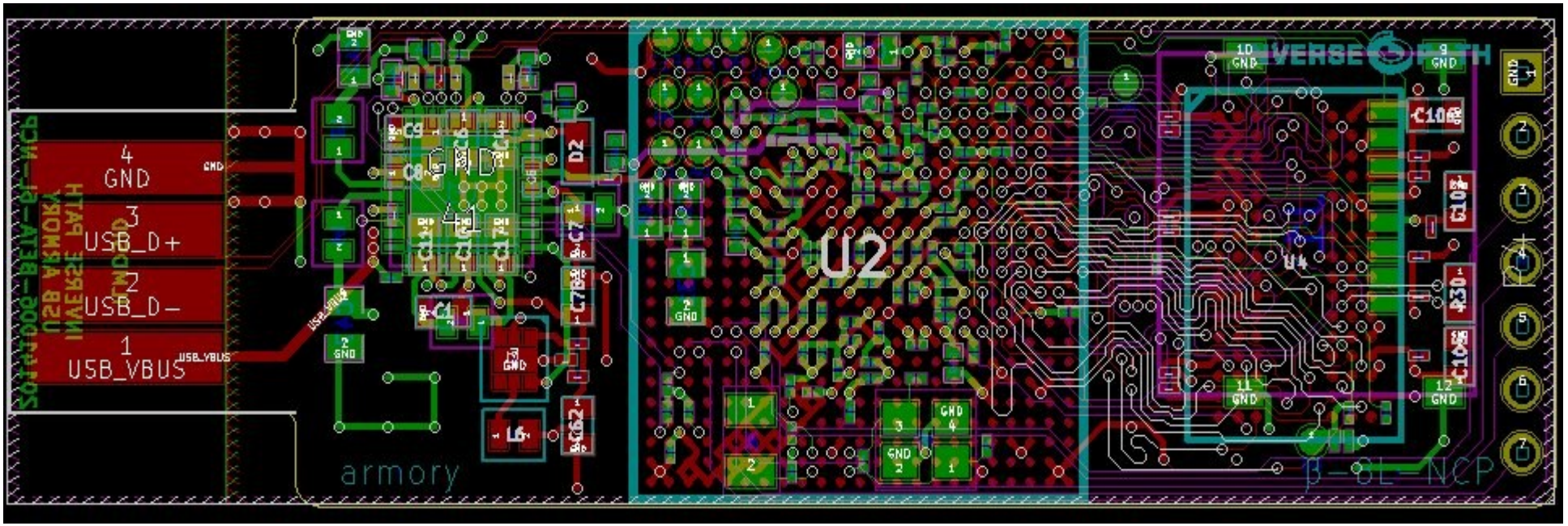


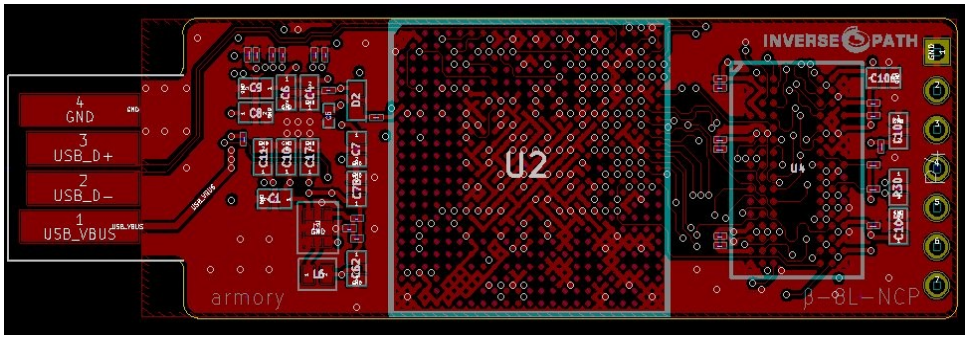


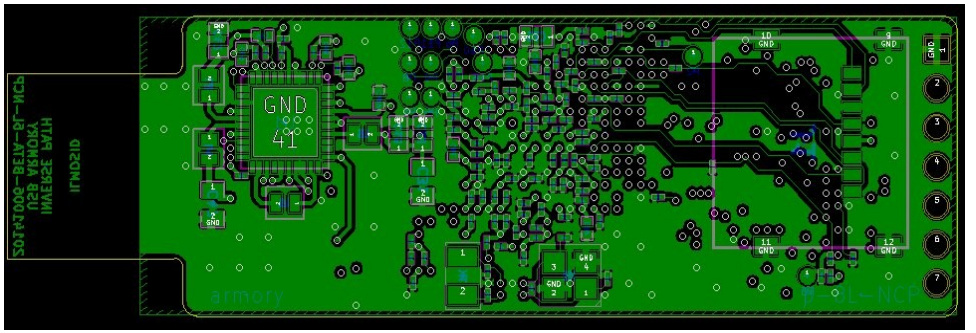
INVERSE PATH

Forging the USB armory

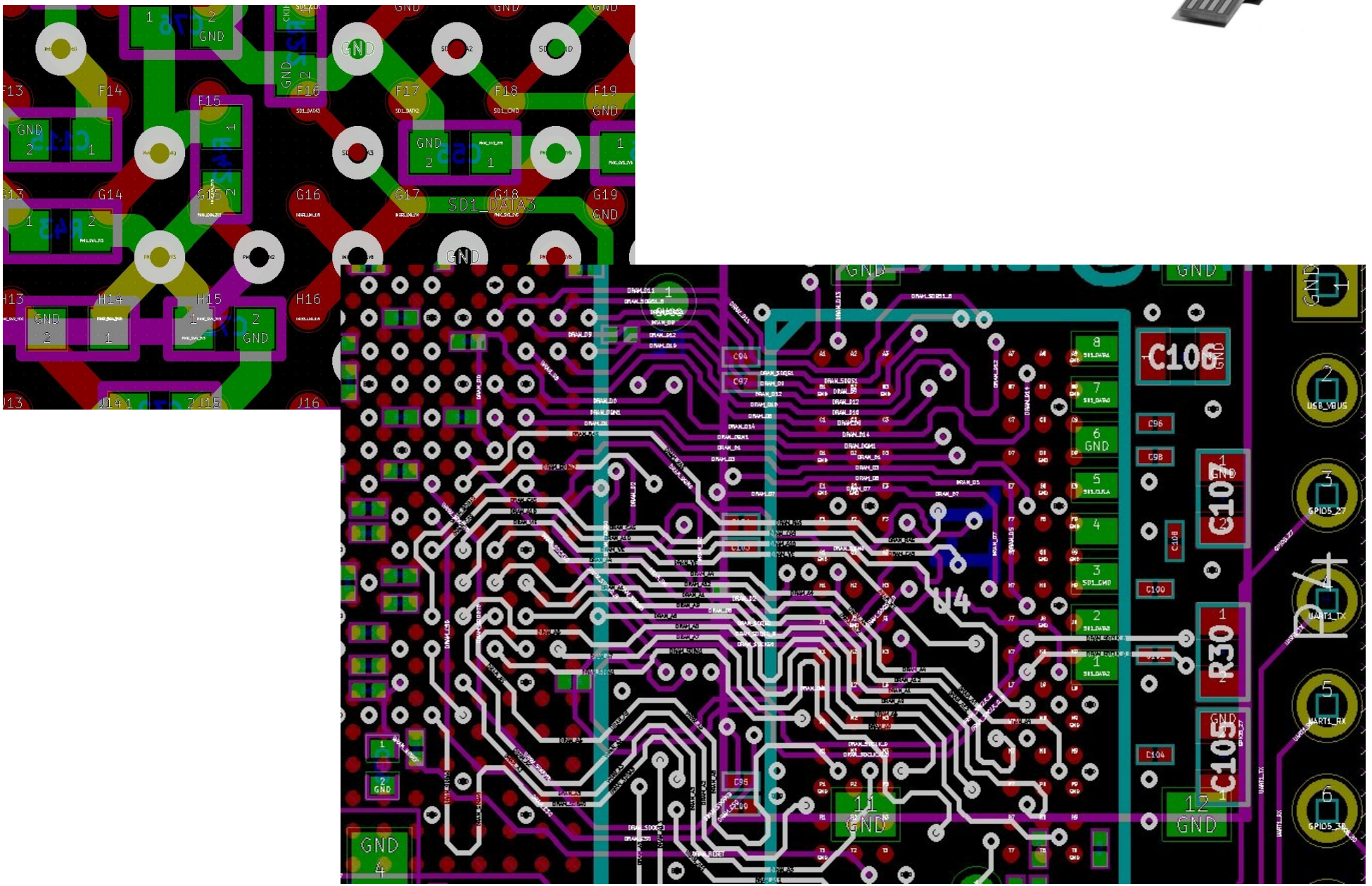
INVERSE PATH



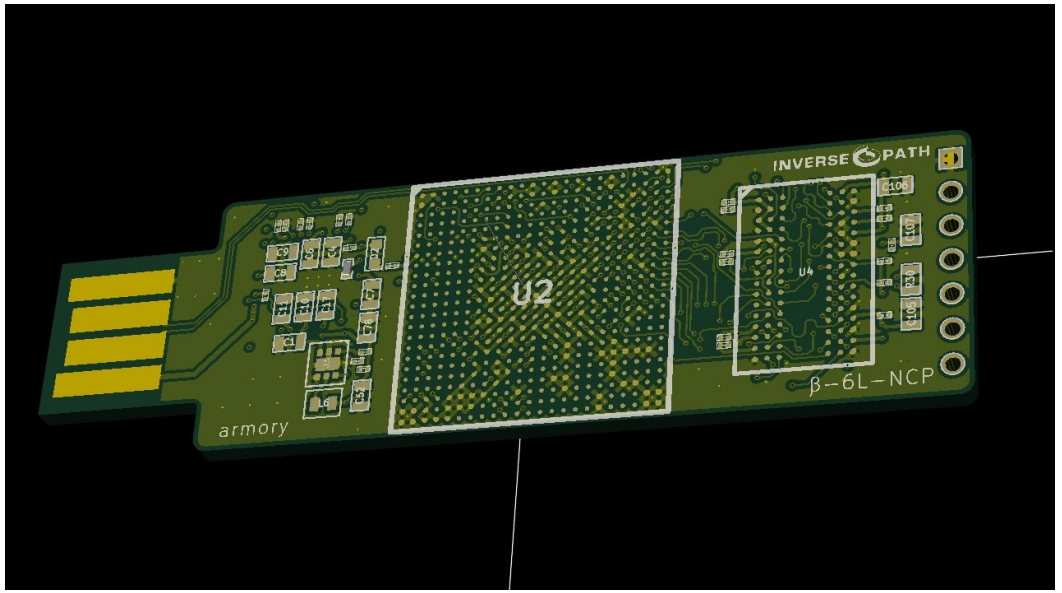


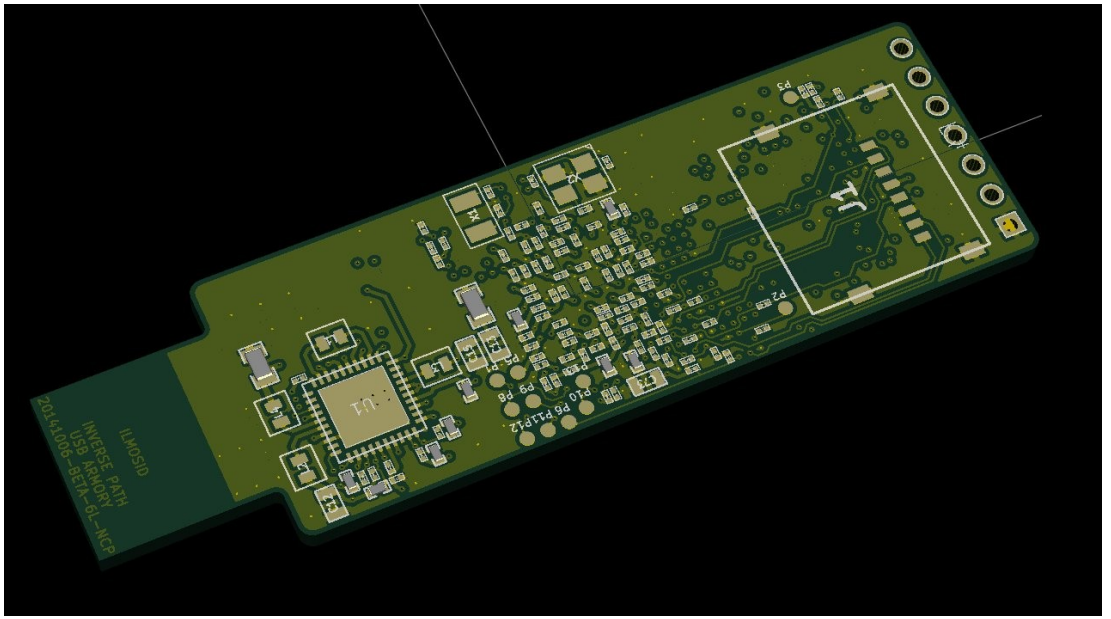


INVERSE PATH

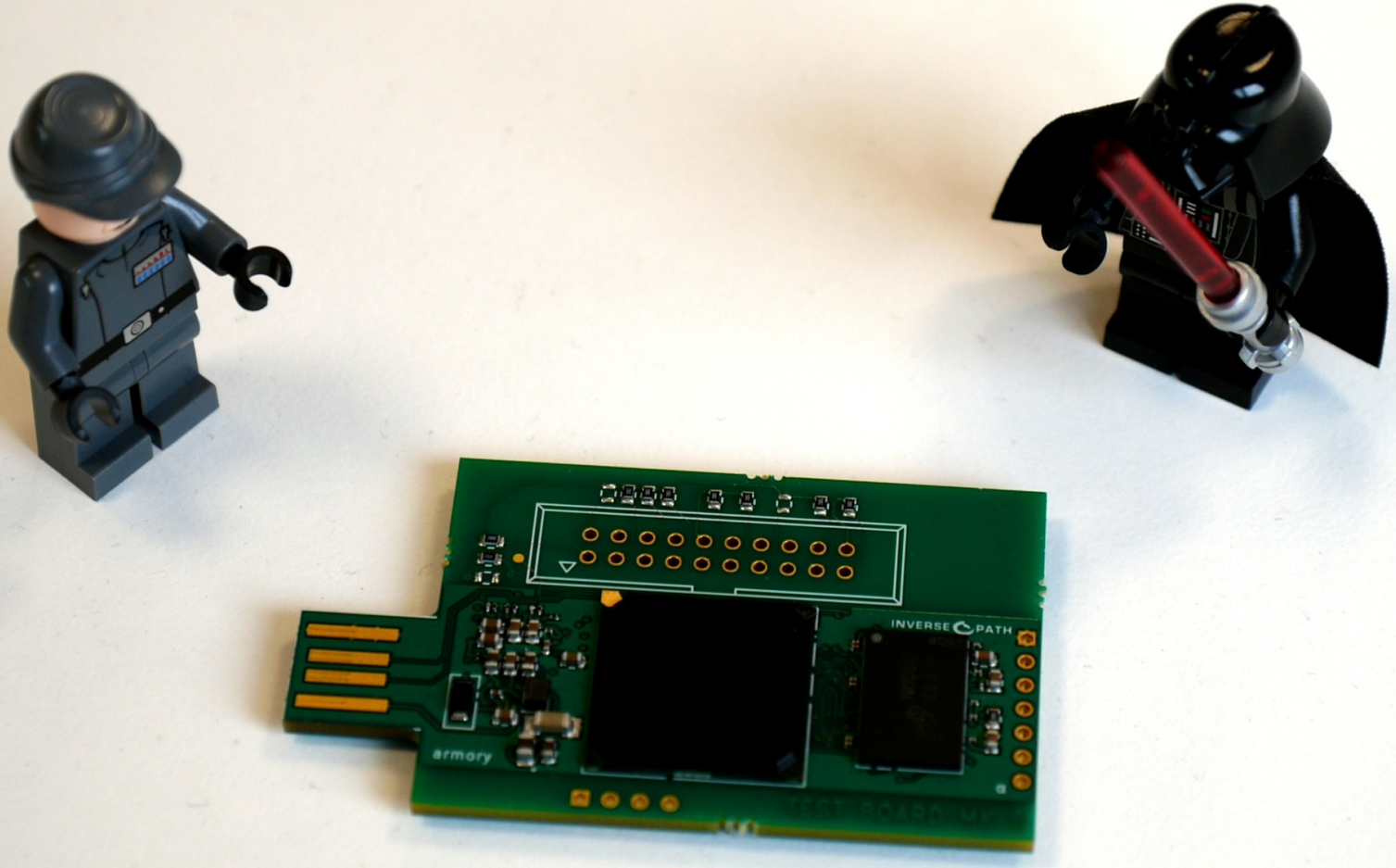


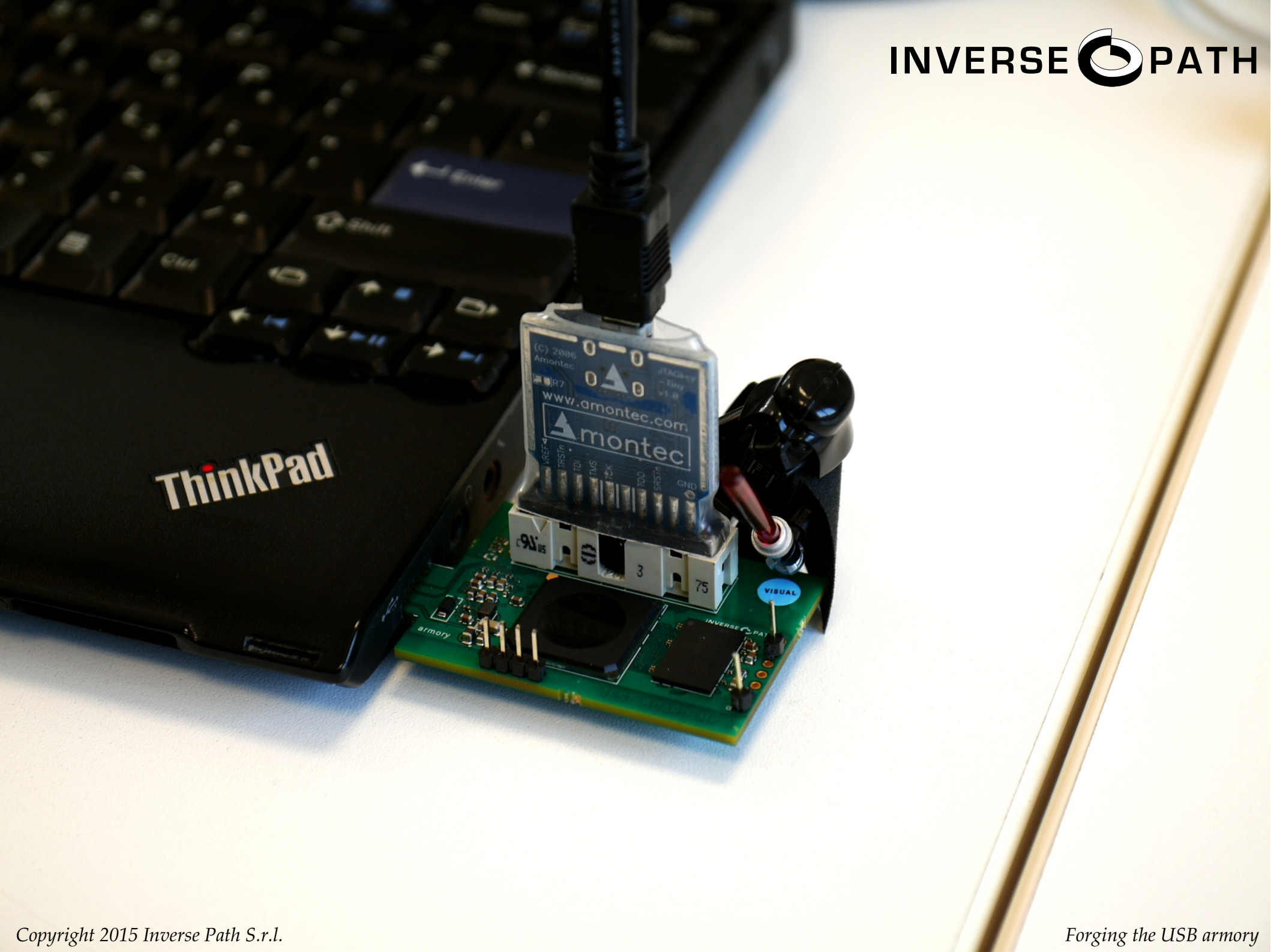
INVERSE PATH





INVERSE PATH





INVERSE PATH

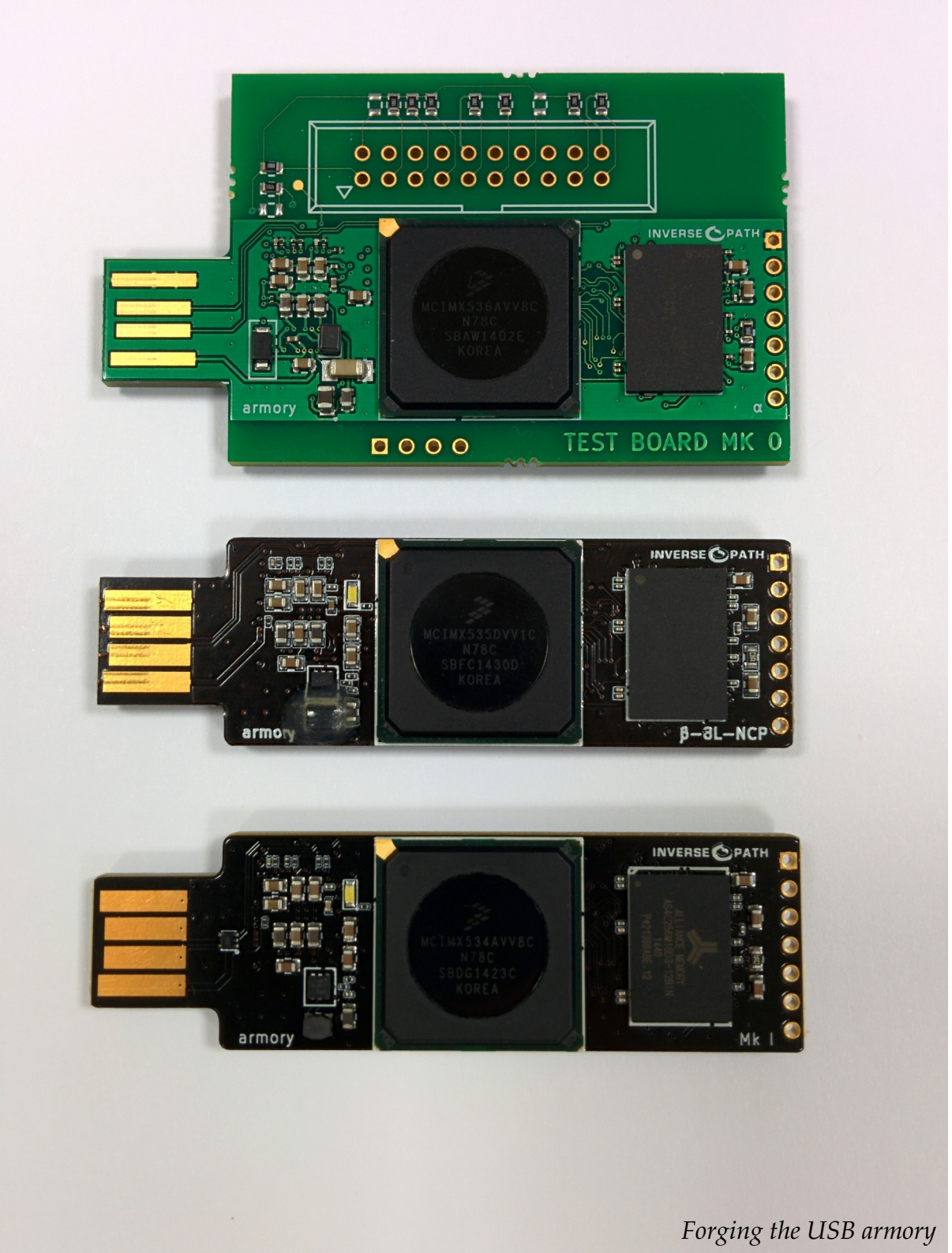
α

βs

8L-NOUSBH,8L, 8L-DDR-LDO, 8L-DDR-NCP

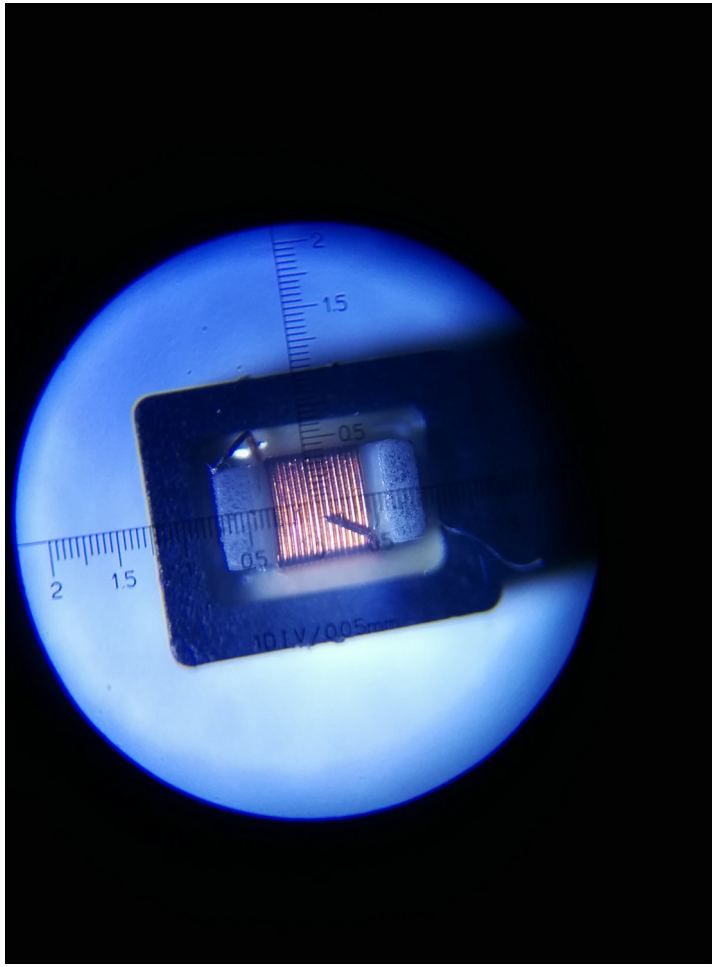
6L, 6L-DDR-LDO, 6L-DDR-NCP

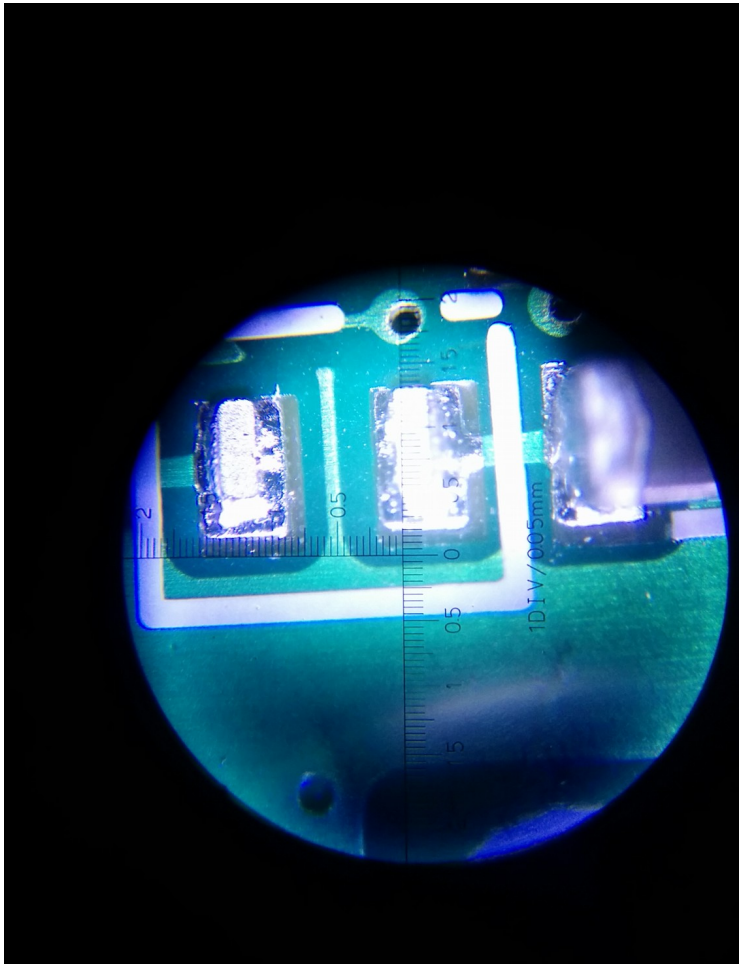
Mk I



Forging the USB armory

INVERSE PATH

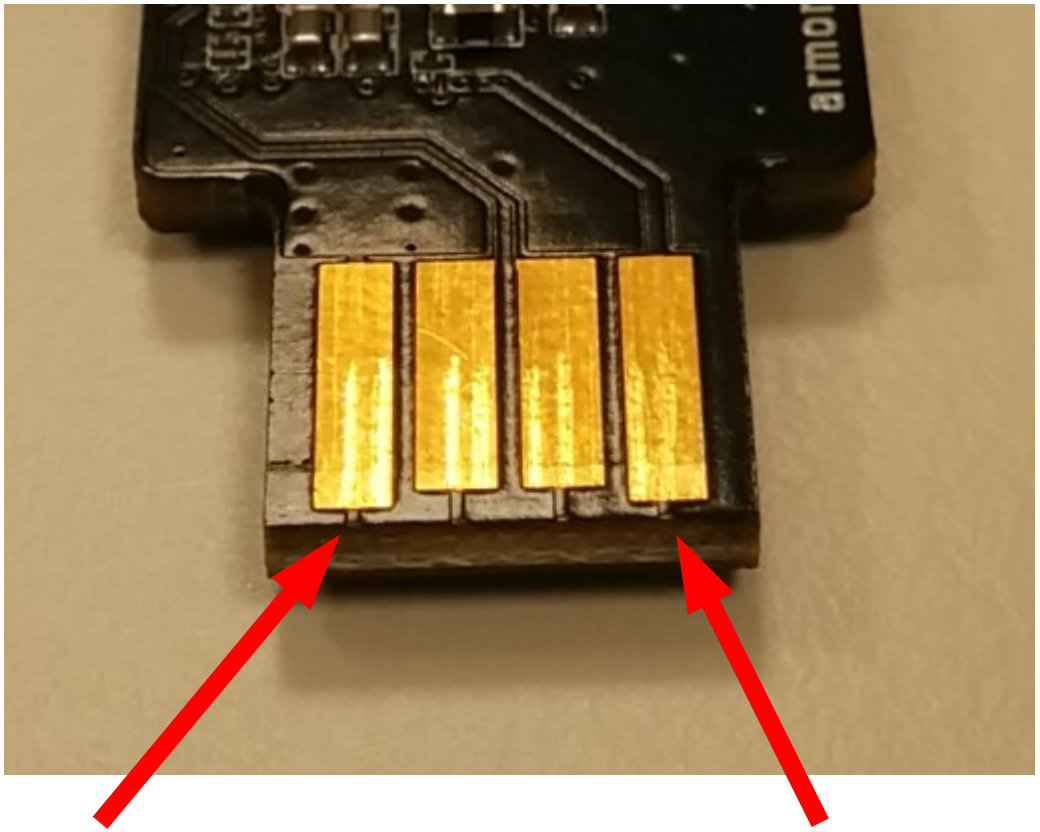


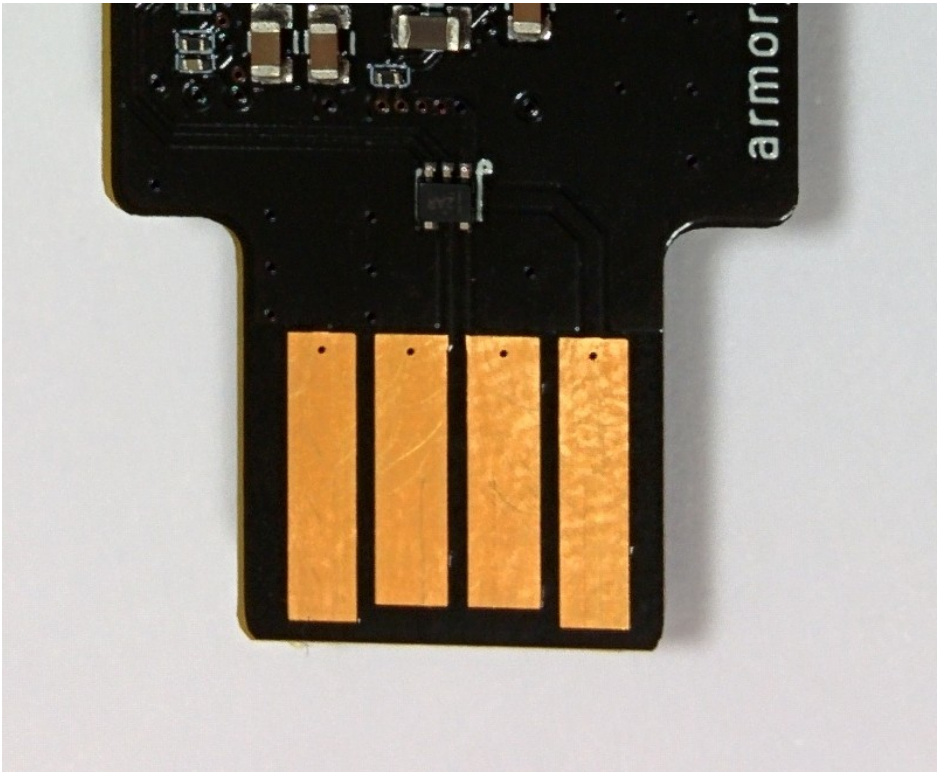


lessons learned #1

tiny inductors are fragile

INVERSE PATH



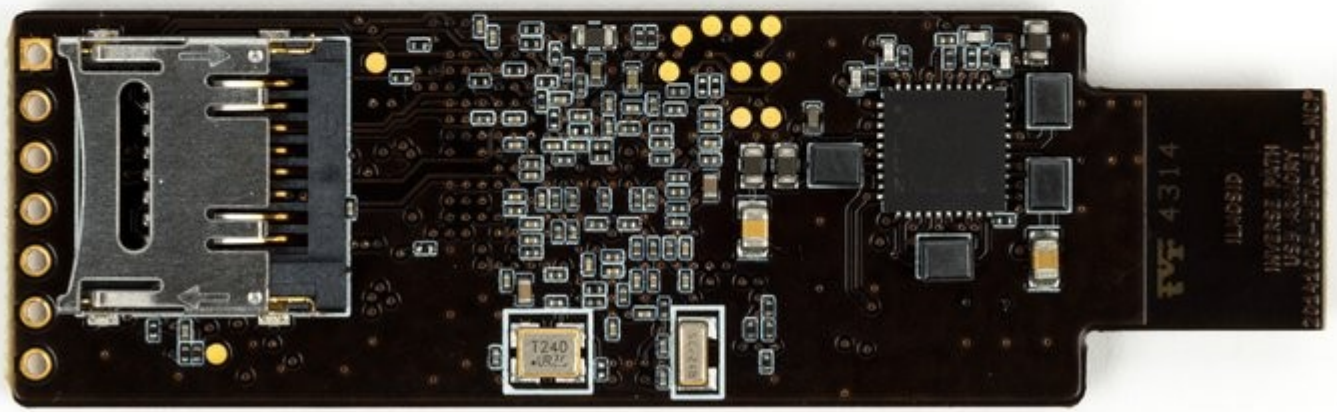


evil

good

lessons learned #2 (the five-second rule)

gold plating traces cause under-voltage on hot swap



TrustZone Support (Genode)

Compiling and running Genode OS (>= 15.02):

```
git clone https://github.com/genodelabs/genode
cd genode

./tool/create_builddir hw_usb_armory
cd build/hw_usb_armory

# in etc/build.conf add "--include image/uboot" to RUN_OPT

make run/tz_vmm
cp var/run/tz_vmm/uImage $SD_CARD_MNT

uboot> ext2load mmc 0:1 0x70200000 /boot/uImage-genode; bootm 0x70200000
```

Requires minimally patch Normal world kernel compiled as follows:

```
make ARCH=arm zImage LOADADDR=0x80008000 modules
```

Secure Mode Monitor (LED example)

```
@ set GPIO4 to SECURE
        movw    r0, #0x33
        movt    r0, #0xff
        ldr     r1, =CSU_CSL
        add     r1, r1, #4      @ CSL1
        str     r0, [r1]

@ set IOMUXC to SECURE
        movw    r0, #0x33
        movt    r0, #0xff
        ldr     r1, =CSU_CSL
        add     r1, r1, #20     @ CSL5
        str     r0, [r1]

@ set OCRAM to SECURE
...
_secure_monitor:
        mov     r10, #0xcafe
        cmp     r0, r10
        beq     smc_handler
        beq     to_nonsecure
```

Secure Mode Monitor (LED example)

```
smc_handler:

        ldr     r10, =IOMUX_LED
        mov     r0, #1
        movt    r0, #0
        str     r0, [r10]               @ set the pad to GPIO

        ldr     r10, =GPIO4_DIR
        movw    r0, #0xffff
        movt    r0, #0xffff
        str     r0, [r10]               @ set direction to output

        ldr     r10, =GPIO4_DR
        ldr     r0, [r10]
        mvn     r0, r0
        str     r0, [r10]               @ toggle LED output

        movs    pc, lr
```

TrustZone Support (Direct)

Secure Mode Monitor (LED example)

```
static int beg_for_led_switch(void)
{
        printk("dear smc, kindly switch the LED\n");

        /* give control to the secure monitor */
        asm volatile ("movw r0, #0xcafe");
        asm volatile ("smc #0");

        return 0;
}
```

The LED is hardware restricted via TrustZone to Secure monitor control.

A trivial interface implementation between Nonsecure Linux and Secure monitor illustrates a simple request for LED switching.

INVERSE PATH

INTERLOCK

http://github.com/inversepath/interlock

Open source file encryption front-end developed, but not limited to, usage with the USB armory.

Provides a web accessible file manager to unlock/lock LUKS encrypted partition and perform additional symmetric/asymmetric encryption on stored files.

Take advantage of disposable passwords, “nuking” option.

Design Goals

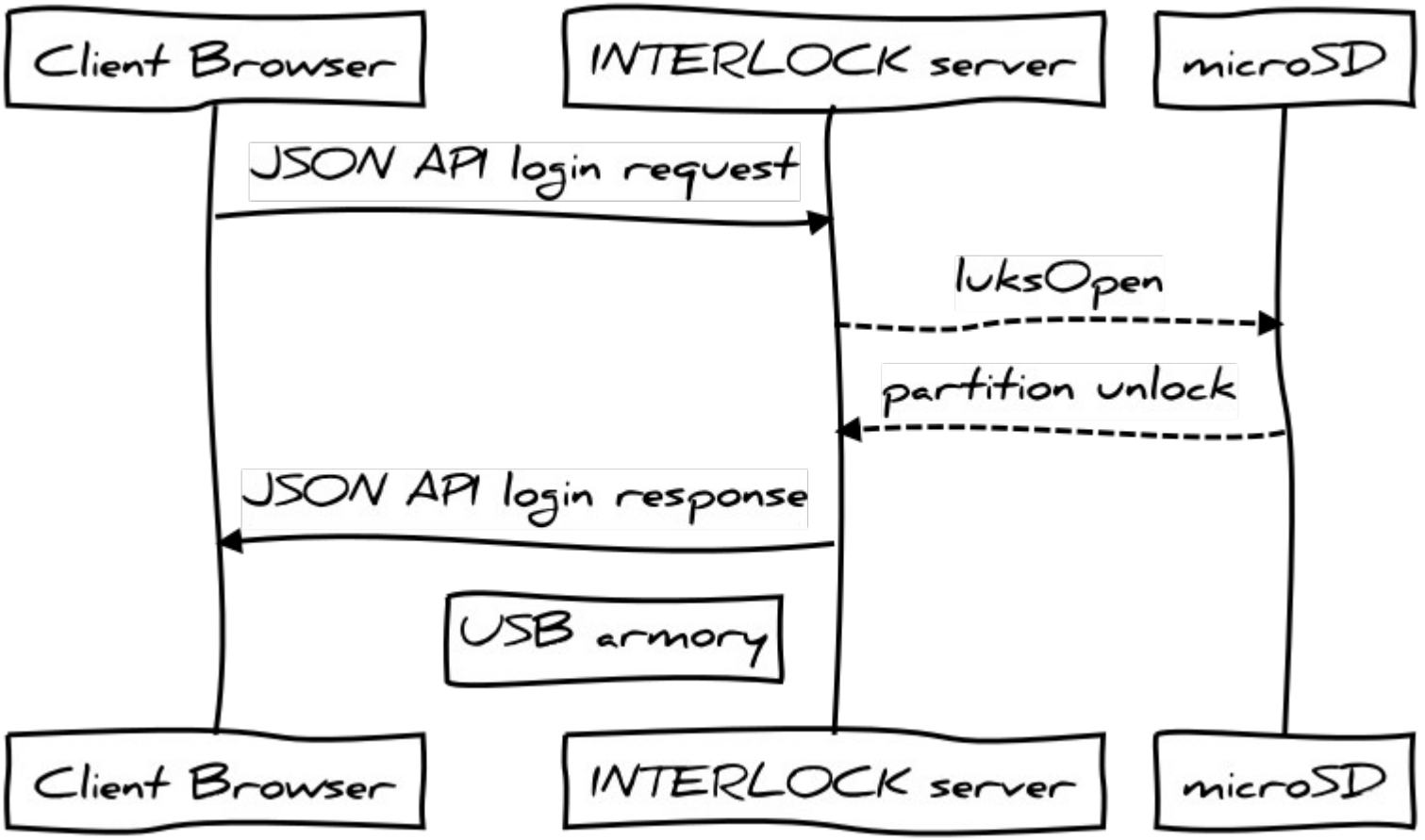
Clear separation between presentation and server layer to ease auditability and integration.

Minimum amount of external dependencies and footprint.

Encrypted volumes: LUKS encrypted partitions
Asymmetric ciphers: OpenPGP
Symmetric ciphers: AES-256-OFB w/ PBKDF2 + HMAC
Security tokens: Time-based One-Time Password Algorithm (Google Authenticator)

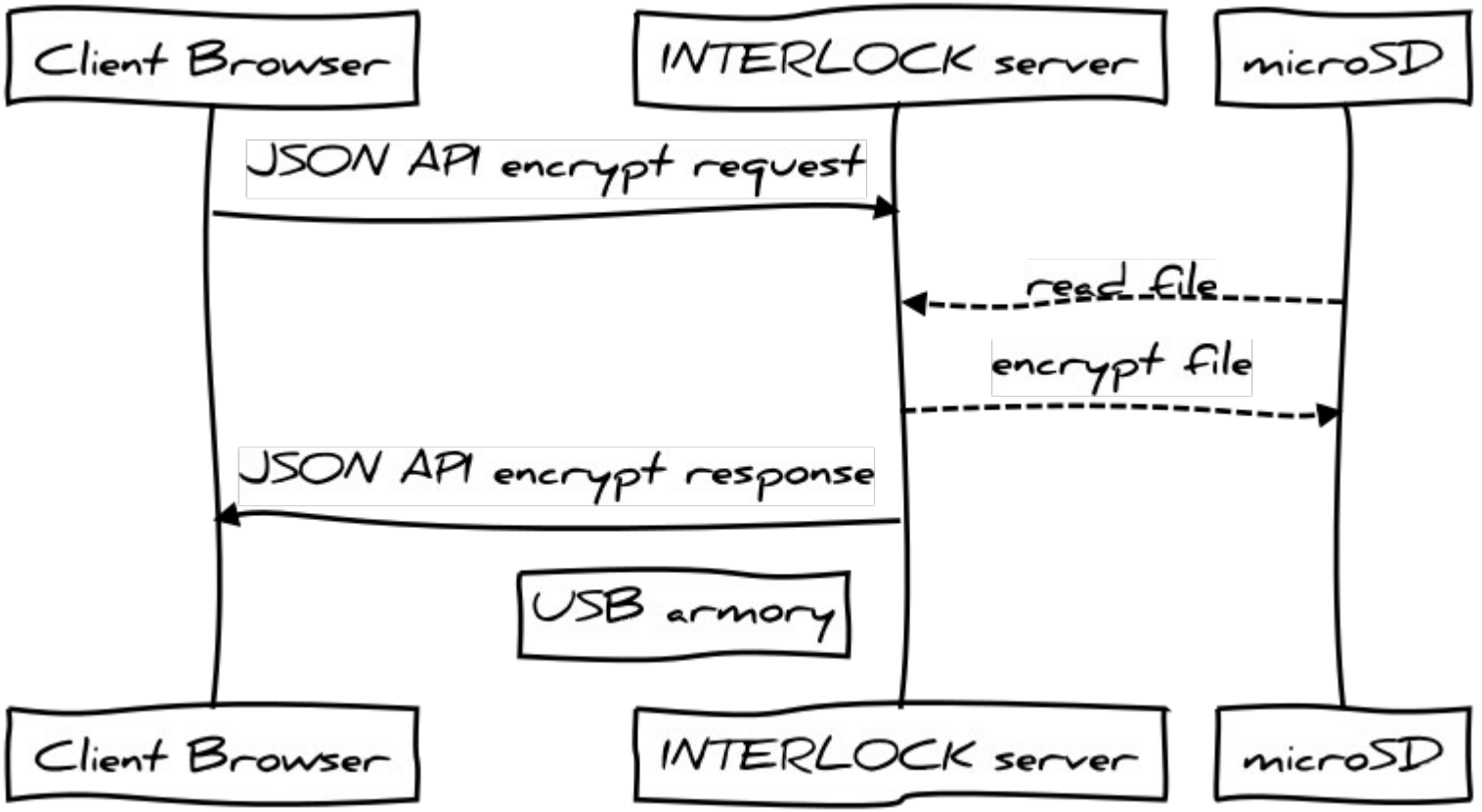
INVERSE PATH

Authentication credentials are directly tied to LUKS partition.



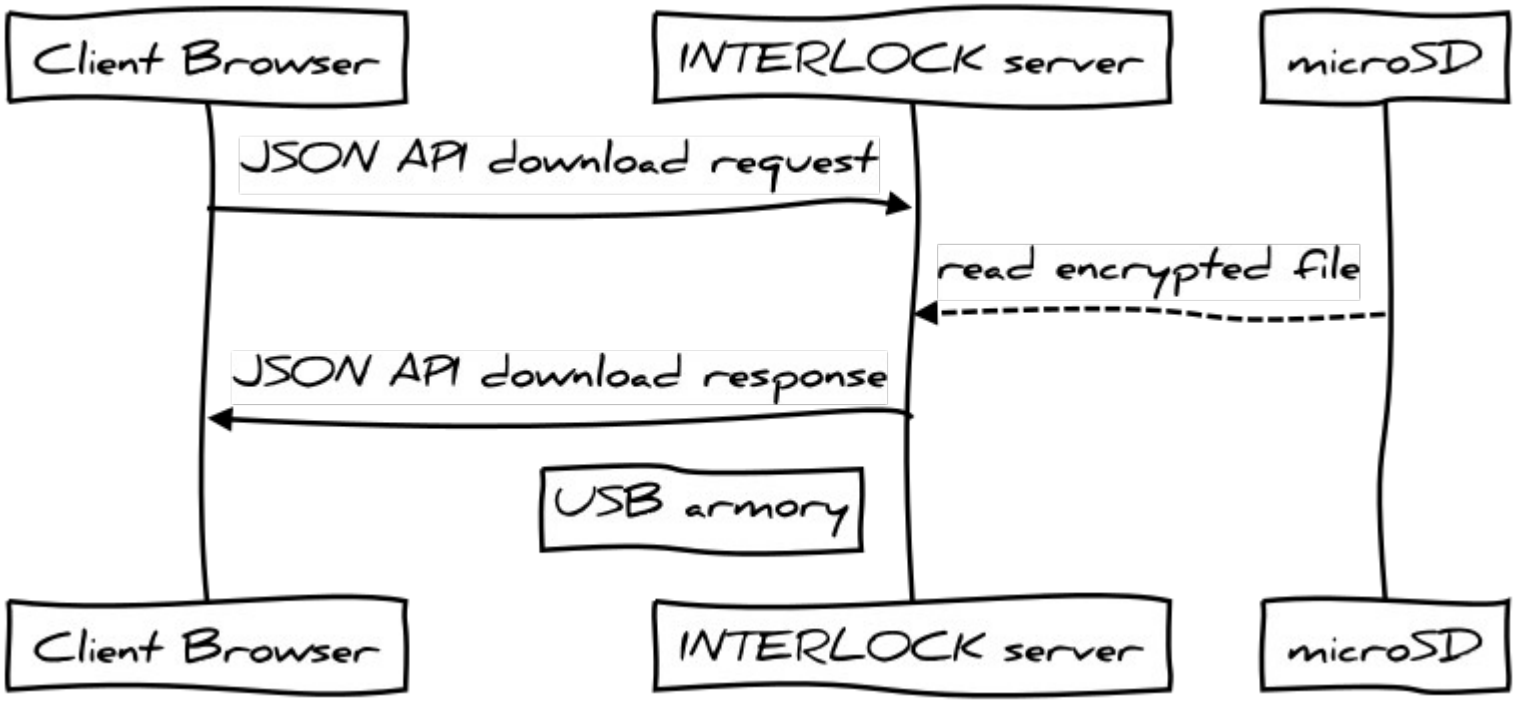
INVERSE PATH

Files can be further encrypted on the USB armory...



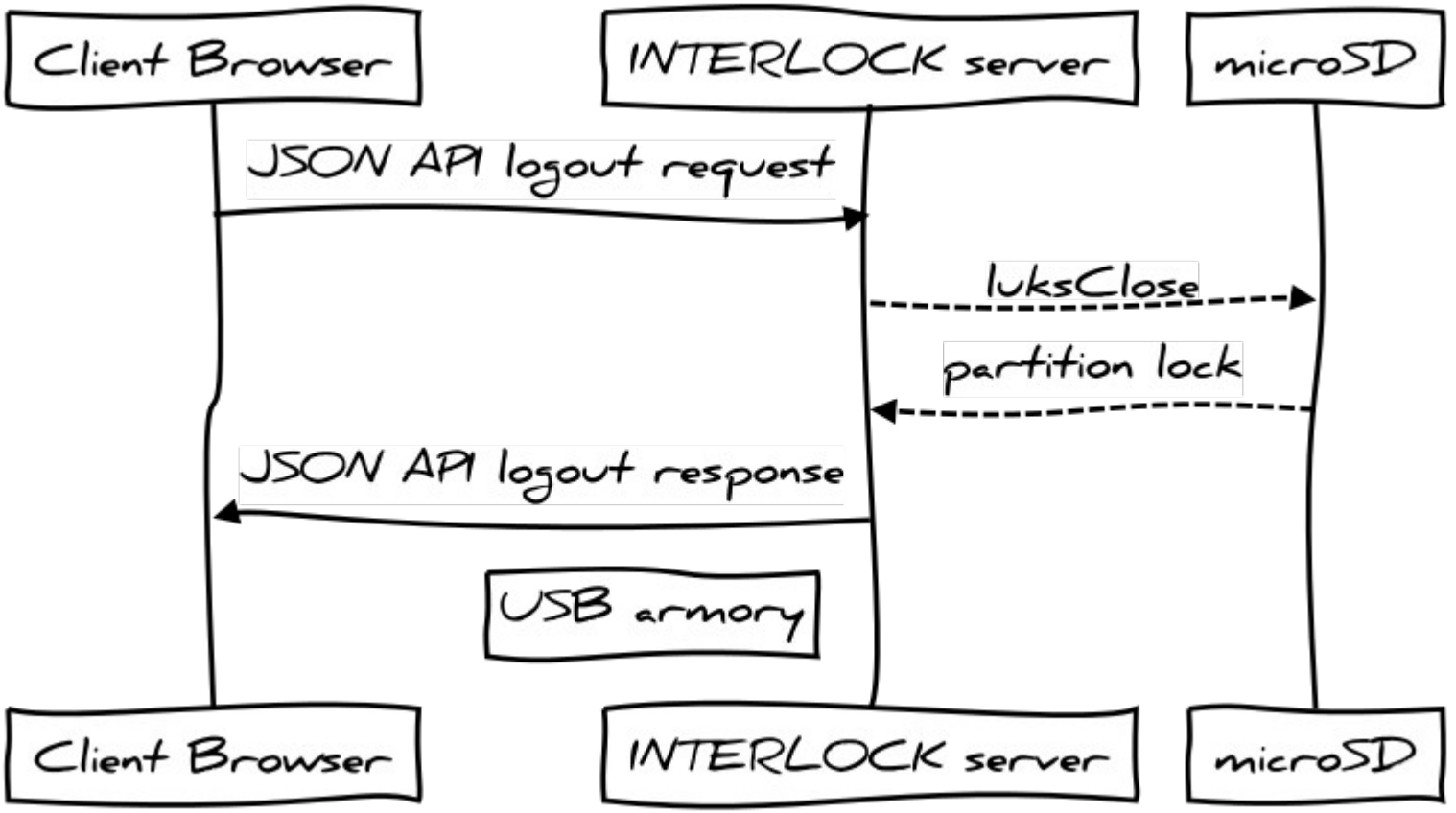
INVERSE PATH

…and later downloaded.



INVERSE PATH

Logging out locks the encrypted partition.



INVERSE PATH

Thank you!

Q & A

Andrea Barisani

<andrea@inversepath.com>